UiO : Department of Informatics
University of Oslo

Linked Open Data Utilization in a Major Digital News Publisher

Trine Frimannslund

Master's Thesis Spring 2015

Linked Open Data Utilization in a Major Digital News Publisher

Trine Frimannslund

4th May 2015

Abstract

Today the World Wide Web contains a vast amount of information, available to us mainly through HTML-documents viewed through a web browser. Linked Open Data is a field aiming to collect and present some of the information available on the Web in a machine-readable way. Among many goals is to be able to unite the various data available today in different formats, to extract even more data from it, and aid content retrieval.

This Master's thesis explore how Linked Open Data can be used in one of Norway's biggest online news publishers, Verdens Gang (VG). I've also developed functionality for extracting Linked Open Data to assist the journalist with supplemental information in a story as it's being written. The functionality was implemented as a plug-in in their publishing system.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to take this opportunity to thank Gisle Hannemyr for his guidance as my main supervisor. Gisle has continuously offered his expertise, along with the occasional anecdote, which has been invaluable throughout this process.

I've received great help from VG along the way, most of all from my external supervisor Tommy Jocumsen. Tommy has expressed deep interest in the work I've done, and motivated me to do my best, for which I am immensely grateful. I would also like to thank Kristoffer Brabrand and André Roaldseth from the development department for all their patience and advice, not to mention help regarding technical issues.

A special thanks to editorial manager Tor-Erling Thømt Ruud and the journalists for taking time out of their busy day to help me. Everyone in the editorial department were always happy to participate, which made my role as a Master's student and researcher much more enjoyable.

Finally, I would like to thank my family and friends for their love and support along the way.

Chapter 1

Introduction

1.1 Background for the thesis

This thesis is done on the background of a wish from VG to explore the opportunities of Linked Open Data. The definition of Linked Open Data will be provided in part 2.3. It was initially intended to serve as inspiration for a potential side-project for the VG developers, but happened to be a subject that I'm personally interested in, and suitable for a Master's thesis.

The thesis is done purely as a Master's student, although the information and contacts I've gathered as a part time employee at VG has certainly helped along the way.

1.2 About VG

Verdens Gang, best known by its initials VG, is currently Norway's most read newspaper of all time[1]. With more than 1 million online readers every day, their website vg.no is the most popular website in Norway. In 1966 the company was bought by Schibsted, joining multiple other newspapers like Fædrelandsvennen, Bergens Tidende and the swedish Aftonbladet[2]. VG does not only produce news, but also has different subsites, like Vektklubb, MinMote, VG Live, VGD, TV-guide, Pent, VGTV, Godt, VG-lista. Only a few of these are directly related to news.

1.3 Problem area

There has been increasing interest in Linked Open Data in the media industry the past years. One of the first newspapers to explore this was the BBC, one of the world's largest and oldest broadcasting companies[3]. In 2007 they published their first vocabulary, comprised of semantic data on BBC programs. These were linked to the semantic Wikipedia, called

[1] http://www.vg.no/nyheter/innenriks/media/vg-nett-mest-lest-noensinne/a/546528/, viewed 4 May 2015

[2] http://www.schibsted.com/en/Media-Houses/, viewed 4 May 2015

[3] http://en.wikipedia.org/wiki/BBC, viewed 4 May 2015

DBpedia (more on this later), and thus became part of the vast collection of Linked Open Data available. The Guardian and NY Times followed a few years later with other variations of Linked Open Data use that are explained further in part 2.4. In spite of these explorations, Linked Open Data remains unknown to most media and news agencies. Although this is not surprising given its novelty, researching new ways of conveying, organizing and researching for news may prove valuable to the industry, especially during this transformatory stage the industry is in.

For the past years there has been an increasing demand for the news industry to evolve. This is largely due to the rise of digital journalism, a concept that has only expanded since it emerged in the late 1990s. Not only does everyone have access to the Internet, but the access is constant and via multiple, heterogeneous devices. Furthermore, websites like Buzzfeed and the Huffington Post are earning an increasing amount of revenue and popularity by discovering new ways of utilizing the web to collect and distribute content. The Huffington Post launched "Off the bus" in 2008, a project crowdsourcing news stories from ordinary women and men[4]. Additionally, the Huffington Post is purely an online newspaper, meaning they are free from the distribution costs of regular newspapers. "Viral" websites like Upworthy, Buzzfeed and Buzzit are relying heavily on social media presence, and reusing content from other sites. The result is a huge amount of traffic, and massive online popularity.

In 2013 a report from the NY Times was leaked[5], containing recommendations for which measures should be taken in order for the company to remain relevant moving forward. One of the issues discussed was the challenge of keeping their online readers on the NY Times site, i.e. avoiding that readers visit external sites like Wikipedia for additional or contextual information around an article. One of the solutions suggested was republishing "evergreen" content, which is content that always remains relevant. Page 26 of the report reads:

> We need to think more about resurfacing evergreen content, organizing and packaging our work in more useful ways and pushing relevant content to readers. And to power these efforts, we should invest more in the unglamorous but essential work of tagging and structuring data.

NY Times produces around 300 URLs every day, resulting in an enormous database of documents containing a huge amount of information. Reusing older news content can prevent them from producing similar information twice - a great advantage for any business. The technical challenge lies in recognizing older articles that can provide additional value to a given news item, and presenting it to the reader in a sensible manner. Page 28 in the leaked report quotes editor in-chief at Vox.com Ezra Klein:

[4]http://www.huffingtonpost.com/howard-fineman/offthebus-huffington-post_b_-891921.html, viewed 4 May 2015

[5]http://www.niemanlab.org/2014/05/the-leaked-new-york-times-innovation-report-is-one-of-the-key-documents-of-this-media-age/, viewed 4 May 2015

> Journalists are better than ever at telling people what's happening, but not nearly good enough at giving them the crucial contextual information necessary to understand what's happened

Providing context can be achieved through resurfacing evergreen content, as mentioned, but also through importing content from other sites. The technical challenge to this relates to finding valuable data in an appropriate format that can easily be imported. Both of these represents ways Linked Open Data can provide value, and are outlined further in chapter 3.

1.4 Research questions

The aim of this thesis is to explore how one major digital news publisher, namely VG, can utilize Linked Open Data. By Linked Open Data I mean data that is part of the Linked Open Data cloud, presented in section 2.3.

As part of my thesis I've developed functionality that demonstrates one of the ways Linked Open Data can be used. The functionality is implemented as a plug-in in VG's publishing system called DrPublish.

My research questions are as follows:

- What are the challenges of utilizing Linked Open Data in the plug-in?
- To what extent do journalists experience the plug-in as useful?
- How do the journalists experience researching using the plug-in compared to traditional researching?

1.5 Structure of the thesis

Chapter 2 - Related work gives an overview of the Semantic Web standards needed to understand the rest of the thesis, as well as related research on the news industry and Linked Open Data.

Chapter 3 - Suggestions for use of Linked Open Data goes into the various ways VG can utilize Linked Open Data.

Chapter 4 - Methods and methodology outlines the methods I've used for data gathering and analysis.

Chapter 5 - Prototype presents the prototype I developed as a part of this Master's thesis, including information on the development process, wireframes and more technical aspects.

Chapter 6 - Findings provides the findings from three separate rounds of data collection; the exploratory research and two usability tests.

Chapter 7 - Discussion discusses the results from the previous research.

Chapter 8 - Conclusion is the conclusion to my thesis, which includes some experiences in writing for VG, and my suggestions for the road ahead.

Chapter 2

Related work

> Time flies. It's actually almost 20 years ago when I wanted to reframe the way we use information, the way we work together: I invented the World Wide Web. Now, 20 years on (...), I want to ask your help in a new reframing

This is how Tim Berners-Lee began his talk about the Semantic Web at a TED-conference in February 2009. It was first introduced as a concept in May 2001 when he, along with James Hendler and Ora Lassila, published an article in the Scientific American called "The Semantic Web"(Berners-Lee, Hendler and Lassila 2001). This is where they officially introduced the concept of semantically linked data, ultimately forming what they coined the *Semantic Web*. It has since been the focal point of vast amount of research papers, and Semantic Web standards constitutes a large part of this chapter as it is closely related to the concept of Linked Open Data.

This chapter will give a basic introduction to the Semantic Web and some of its most important components, followed by a brief overview of crowdsourcing, Linked Open Data and some examples of use. The concepts, standards and terms here are by no means exhaustive to the field, but is instead intended to give the reader some insight to what the Semantic Web and Linked Open Data are, what they can do, and what they can mean for the future of the World Wide Web.

2.1 Semantic Web

2.1.1 What is the Semantic Web?

Today the World Wide Web is structured in a way that makes it easy to read for humans. Although it's machine-readable as well, the manner in which the information is structured and represented is not aimed specifically towards machines, which inevitably makes it more cumbersome for a machine to comprehend the meaning behind the data — the semantics beneath.

There are currently multiple different ways of expressing information on the web, each with its own advantages and disadvantages. The issue is the lack of a universal standard, as this would enable merging

information from many different sources. So although information is expressed, one source (e.g. a website) can't easily merge its content with another information source (e.g. an Excel spreadsheet), as the information is expressed in different ways, and there is no automatic way of tying them together. This remains one of the biggest goals of the Semantic Web movement — expressing information using Semantic Web standards as opposed to a proprietary format, so information from vastly different sources can be merged. However, the advantages of this is highly dependent on whether the standard is being used or not. There have been some great achievements in the Semantic Web community the past years, especially with the semantic encoding of Wikipedia, but its future remains difficult to assess during such an early stage.

In Berners-Lee et al.'s article they create a vision of a world where computers can assist humans to a much greater degree than possible today. Tying the information on the web together means having devices that understand what information the user wants and needs, and knows where to get the information and how, which can potentially be a huge leap in the way we interact with technology.

Though Berners-Lee's vision is closely related to Artifical Intelligence, The Semantic Web is also largely about organizing content and using ontologies to do so. Semantic data is placed within ontologies, a term explained in section 2.1.2. The ontology provides the machine with a world view, e.g. that a cocker spaniel is a dog, or in other words - a subclass of dogs. This "world view" is highly valuable as it can be used to understand and organize already existing documents, as well as the information within them. So while Berners-Lee uses examples related to Artificial Intelligence (AI) that might seem futuristic for most people, the benefits of simply organizing data and using ontologies should not be underestimated.

For individuals with a limited understanding of how the World Wide Web works today, the Semantic Web can be difficult to fully comprehend. Here are some scenarios that might be of assistance:

> Samantha is looking to buy a dress for a party this weekend. She wants it to be either red or black, and machine-washable. She does not want to pay more than 70USD, and the store has to be located within a 5 mile radius from her house. She types these data into a semantic search engine, and it returns 6 possible dresses, including pictures, price, which stores they belong to, along with their address.
>
> Jake has a newly discovered passion for old movies, and loves reading movie blogs. He's reading a blog post on a Doctor Mabuse movie, but he wants to know what other movies the director has worked on. Fortunately, by simply clicking on his name, and a box with facts about Fritz Lang shows up. Jake learns that Lang has directed 47 other movies, including Metropolis from 1927.

These scenarios are just two examples of ways the Semantic Web could

be advantageous to users, although it's the technical aspects behind them that illustrate it the best. These will be delved deeper into in later chapters.

Standardization is one of the vital concepts of the Semantic Web. W3C, or the World Wide Web Consortium, is an organization aiming to provide standards for publishing content on the World Wide Web, and is lead by Tim Berners-Lee and Jeffrey Jaffe.[1] The next sections will provide an overview of the W3C standards for the Semantic Web, after quickly outlining some of the characteristics of Semantic Web technologies.

2.1.2 Characteristics of the Semantic Web

Ontologies

The Oxford Dictionary defines ontology as "the branch of metaphysics dealing with the nature of being"[2], which might sound out of place in the domain of Computer Science. For the Semantic Web, an ontology defines the concepts and the relationship between them, including the constraints they have. The ontology organizes and categorizes your information, creating an *information domain model*. Ontologies make it possible to reason about categories, and this way provides an important part when we want to reason about data (described in the next paragraph). There already exists plenty of ontologies on the web today, and proponents of the Semantic Web encourages developers to build upon these, although making your own from scratch can also be a viable solution.

Vocabularies

The terms *vocabulary* and *ontology* tend to be used interchangeably in the Semantic Web community, because they often serve the same purpose. However, a vocabulary decides the names used in the ontology to refer to entities. The concept of a *controlled vocabulary* can be used to grasp the difference. A controlled vocabulary is the opposite of a *folksonomy*, where the content creators themselves choose the classification and/or categorization scheme of the content (Morville and Rosenfeld 2006). Well-known examples include social media websites like Twitter and Facebook who allow users to tag their content using hashtags. The user is free to use whichever hashtag he or she likes, or make up their own, and there are no restrictions defined. The opposite of a folksonomy is a controlled vocabulary, meaning that the terms used are predefined. This involves an authority deciding which terms should be used, and training other people to use it properly.

There are many advantages of having a good quality controlled vocabulary. Among the most important is to support the organization and categorization of documents, which in turn gives meaning to data on what kind of articles are being consumed and/or produced. This kind of data can be used to provide statistics and make visualizations to

[1] http://www.w3.org/Consortium/

[2] http://www.oxforddictionaries.com/definition/english/ontology

aid decision making processes, produce index pages etc.. Maintaining suitable and efficient categories also supports interlinking of content, as you can reason about what a given document is about and thus know something about its relation to other documents. This information can be used to provide related articles, related terms/tags etc. Another important benefit to having a controlled vocabulary is providing standardized terms. Having standardized terms, and possibly variant terms, not only supports information retrieval (which can also be used to retrieve related articles), but also ensures that we can syntactically tell that two text strings are in fact referring to the same thing.

Vocabularies in a Semantic Web context doesn't differ much - it's the terms used for describing entities and their relationships. And as in normal language, there are many different words that can be used to explain the same thing. The term chosen is often dependent on what the vocabulary is made to describe, as there are different vocabularies for different domains of interest. The popular vocabulary Friend Of A Friend (FOAF)[3] is intended to express information about people and their relationships to other people. A person is called a foaf:Person, but in a vocabulary for biologists, a person might be called bio:Human. We can tie these two vocabularies together by expressing that foaf:Person is the same as bio:Human. If we have a lot of data on people which is expressed as as "foaf:Person"s, and other information on lots of "bio:Human"s, Semantic Web technologies enables us to reason upon even larger datasets.

Triples

Semantic data are stored in knowledge bases, sometimes called triplestores. Triples consist of a subject, a predicate, and an object (e.g., Dog isSubClassOf Animal, or Charlotte isType Person). This resembles the structure of simple sentences in linguistics, with subjects, verbs and objects. The language used for expressing triples is RDF (Resource Description Framework)[4], where the subject is a resource, and the object is either a literal or a resource. The resources all have unique names in the form of URIs (Unique Resource Identifiers), and so does the relationship (in this case the predicate). Using URIs ensures that the item is unique across the web, and this way avoids naming conflicts and ensures that we are referring to the correct resource. Each URI can also be an URL (Unique Resource Locator), URN (Uniform Resource Name) or IRI (Internationalized Resource Identifier).

Reasoning

Having defined an ontology, instances can utilize it. E.g., if all cats are animals (cat is a subclass of animal), and Fluffy is a cat, then Fluffy must be an animal as well. Fluffy inherits and exhibits the qualities of both animals and cats. Subclasses and superclasses are common relationships that are reasoned upon, but there are many other types of relationships

[3] http://xmlns.com/foaf/spec/
[4] http://www.w3.org/RDF/

and constraints that will shape the instances' traits. Typical examples are *equivalence*, *domain*, and *range*. Reasoning is usually done by a software, two popular examples being Pellet[5] and HermiT.[6]

Open World Assumption

When dealing with the Semantic Web, it's important to know some of the logic behind it. The Open World Assumption describes one important aspect of reasoning, and goes for all reasoning on Semantic Web data. This assumption says that when a statement is not found, it doesn't mean that it's false; it's simply not computable. If we have a triple stating that Fluffy (the cat) eats fish, and we were to ask the ontology whether Fluffy eats mice, it would answer that it doesn't know. If we were to use to Closed World Assumption, on the other hand, it would return false. Using this example, the open world assumption makes sense. But imagine that we told another ontology that cats only eat one type of food, and then that cats eat mice. We then merge the two ontologies. At that point the ontology will contain triples saying that cats eat both fish and mice, but only one type of food. As a result, the reasoner will compute that mice is the same thing as fish. These repercussions can be confusing for most people, and is therefore worth to keep in mind.

Challenges

The Semantic Web is currently facing multiple challenges. One of the biggest is the enormous amount of information it holds, and how much it should eventually hold. Already a central job in semantic databases is to eliminate duplicate terms and triples. Doing calculations and reasoning on this much information is a job that requires considerable amounts of power from any engine.

Another problem are vague terms, like *small* and *big*. These kinds of words usually appear as a result of user contributions, but are not suitable for the Semantic Web as they are too open to subjective interpretation. Since the Semantic Web allows anyone to say anything about anything, another challenge is proving that the contributing person is who he says he is. This becomes particularly important once we consider the an additional problem, which is deciding whether you can trust this person or not. If the Semantic Web is to continue to take contributions from people online, which is sometimes called *crowdsourcing*, both of these issues need to be addressed.

2.1.3 Semantic Web standards

The following sections present only brief summaries of some of the various Semantic Web standards. For more in-depth definitions and explanations, see Hitzler, Krötzsch and Rudolph 2009.

[5] http://pellet.owldl.com/
[6] http://hermit-reasoner.com/

RDF and RDFS

RDF stands for Resource Description Framework, and is a W3C standard for describing web resources. The purpose of RDF on the Semantic Web is to represent the triple-structure introduced earlier in a machine-readable manner. As mentioned, all resources and relationships in triples are actually URIs. In RDF, a full triple could look something like this:

```
<http://example.com/MomsPizza> <http://example.com/hasIngredient>
    ‘‘1 cup cheese’’ .
```

The URI for Mom's pizza is the resource, and the URI for having ingredients defines the relationship. "1 cup cheese" is a literal. Since these triples can be expanded, it's common to include prefixes for both readability and saving space:

```
@prefix ex: <http://example.com/>.
ex:MomsPizza ex:hasIngredient ‘‘1 cup cheese’’ .
```

Using the Friend Of A Friend (FOAF) vocabulary mentioned earlier, we can express that:

```
@prefix foaf: <http://xmlns.com/foaf/0.1/>.
foaf:Trine rdf:type foaf:Person .
```

It can be useful to note that these examples are written using the Turtle language. RDF offers several additional serialization formats, such as RDF/XML, N3, N-Triples and JSON-LD.

So far we've only dealt with instances. For describing more complex kinds of relations, one option is to use RDFS, or Resource Description Framework Schema.[7] RDFS follows a more object-oriented approach than RDF. It allows you to define classes, properties, subclasses, subproperties, domains, ranges and much more, but is written in the exact same syntax as RDF. The goal is to ease the combining and merging of different datasets by describing groups of datasets, rather than individual instances. An example of RDFS would be:

```
ex:Sneakers rdfs:subClassOf ex:Shoe
```

OWL

OWL[8], or the Web Ontology Language, was specifically built for situations where machines need to process the information, not merely display it to humans. For that reason, OWL both expresses meaning about the data, and functions in a way that eases the job for reasoners. The meaning that OWL can express bares resemblance to RDFS, but it has a much larger vocabulary and can be written in other languages than RDF. Furthermore, OWL allows you to make links across databases, like:

[7] http://www.w3.org/TR/rdf-schema/
[8] http://www.w3.org/2001/sw/wiki/OWL

```
foaf:Pete owl:sameAs ex:Pete
```

This triple says that two resources (foaf:Pete and ex:Pete), although having different URIs, are actually the same. In other words; all other triples that includes foaf:Pete also goes for ex:Pete and vice versa. This is something that could typically be necessary when merging two distinct databases.

SPARQL

SPARQL[9], or SPARQL Protocol and RDF Query Language, is the W3C query language standard used for retrieving semantic data coded in RDF(S). Its syntax is much like traditional SQL variations, an example being:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?email
WHERE {
    ?person foaf:name "Charlotte" .
    ?person foaf:email ?email
}
```

In short, this query retrieves Charlotte's email adress. If we were to explain in greater detail, one could say that we are retrieving the string-value that has the foaf:email-relationship to any resource who has "Charlotte" as the object in a foaf:name-relationship. This query will return a single column named "email" with all the e-mail adresses connected to the people in the FOAF-database with the name "Charlotte". As with RDF(S), it's common to use prefixes to minimize the query. This is done by using the PREFIX-keyword. Other keywords available are OPTIONAL, UNION, FILTER, REGEX and DATATYPE. In the above example we used a SELECT-statement, but other result formats are also available, some of the most common being CONSTRUCT, DESCRIBE and ASK.

2.2 Crowdsourcing

2.2.1 What is crowdsourcing?

Crowdsourcing, according to Jeff Howe (Howe 2008), is when you take a task traditionally performed by a designated agent and outsource it by giving it to a large and undefined group of people. In other words, it's a form of delegation where the crowd is now performing the job that used to be done by only a selected few.

In his book *Crowdsourcing: Why the power of the crowd is driving the future of business*, Howe relates four developments to the rise of crowdsourcing. The first is what he terms "the renaissance of amateurism", which is the

[9] http://www.w3.org/TR/sparql11-overview/

trend rising where more and more people do activities that were previously only done by professionals. An example is making videos. Previously filming equipment was so expensive that only professionals had it. Today the costs of filming equipment has been reduced by so much that almost everyone with a smartphone now carries a form of video taping device in it. In addition, high-quality filming equipment is available in stores and online. By being able to involve amateurs, companies today can tap an enormous source of whatever they need without having to pay professionals. Furthermore, it allows for people without a professional degree to contribute their work, potentially creating a win-win situation for all parts.

The second development Howe describes is the rise of open source projects. There exists multiple examples in which this development approach has been successful, perhaps some of the most well-known being Wikipedia and the Linux operating system. Open source projects are projects where the source code of computer software is made available online for people to look at, modify, copy and contribute to, and was a revolutionary idea in the beginning. Since then it has only become increasingly popular.

Another contributing element is the reduction in cost of producting and distributing content. Furthermore, it has become easier to find information on how to do these things, and the user interfaces of the relevant software has become much more user friendly.

And finally, the World Wide Web has provided people in vastly different geographical locations to come together through a shared interest. These kinds of online group or communities can be hugely advantageous both for the company and for the members. The company doesn't have to employ administration staff, because the communities are often self-regulatory, and the members of the community have a place to share their work and get feedback.

2.2.2 Crowdsourcing today

Further on in his book, Howe presents four main use areas where crowdsourcing is prevalent today. The first is the use and application of collective knowledge, which he explains by referring to the Diversity Trumps Ability Theorem put forward by Scott E. Page in his book "The difference: How the power of diversity creates better groups, firms, school and societies" (2007). This theorem states that a collection of problem solvers collected randomly outperforms a collection of the best individual problem solvers. Companies have used collective knowledge when searching for new ideas (like asking the customers to come up with a new product idea and offering a reward) or predicting events.

Another way of crowdsourcing is collecting user generated content, a well-known example being Wikipedia, which we will get back to in the following section. Other examples include reality TV shows like Idol, where the crowd supplies the talent and entertainment.

Crowds can also filter and organize large amounts of information.

Again we can exemplify by using Idol, in which the crowd votes on who they think is the best. Rating videos on YouTube and other similar platforms are other ways in which the collective decides what is good content and what is not. Additionally, crowds can fund projects. This phenomenon has its own term, "crowdfunding", and has been a successful approach to collecting capital in many instances (Gerber and Hui 2013).

2.3 Linked Open Data

As explained in chapter 1, the main objective of this Master's thesis is to explore how VG can utilize Linked Open Data. The above sections have delved deeper into the Semantic Web and its standards, which can be helpful in understanding Linked Open Data.

Linked Open Data is a combination of two concepts: Open Data refers to the idea that data should be free for anyone to use[10]. Used in Linked Open Data-terms, it means data published under an open licence[11]. Linked data is merely data that is linked through a machine-readable language. However, in order to be useful to other actors, it should also be open. Thus the term Linked Open Data.

Tim Berners-Lee uses five stars to illustrate different levels of data. For the first star, your data has to be on the web, published under an open licence. For two stars, it should be machine-readable data, like an Excel spreadsheet. For three stars, make it a non-proprietary format, like CSV. Non-proprietary means that no-one has licensed it (for instance is Excel owned by Microsoft), and CSV is a file format similar to a standard text file (.txt). In order for other people to link to your data, you should use a W3C open standard language, like RDF. This earns you four stars. The fifth and last star is only given if you also manage to link your data to other people's data.

In order to make your data easy to use for external actors, and potentially reach the five star level, Berners-Lee introduces a set of guidelines or "best practices" for publishing data on the web. His guidelines are as follows:

1. **Use URIs as names for things**

   While you previously might have stored your information as text or string values in a relational database, his suggestion is to use URIs to refer to the various entities instead. As stated earlier, using URIs ensures that the item is unique across the internet, and this way you avoid naming conflicts and ensure that we are referring to the correct resource. If you're not using URIs, it's not Semantic Web. You can use your own vocabularies or an existing one, which is explained further in section 2.4.2. Berners-Lee also wants you to

2. **Use HTTP URIs so that people can look up those names**

[10] http://en.wikipedia.org/wiki/Open_data, viewed 4 May 2015
[11] http://www.w3.org/DesignIssues/LinkedData.html, viewed 4 May 2015

This is another widely accepted rule in the Semantic Web community. Using HTTP ensures that anyone with a web browser can access the URI, making it exponentially more useful to other people. W3C provides multiple resources on how to choose URIs, for instance [12] and [13]. In addition,

3. **When someone looks up a URI, provide useful information, using the standards (RDF*, SPARQL)**

   In practice this means making your server return something useful to the user, like a model of the graph database or something else that gives a better understanding of where he is and what he's looking at. And the last rule relates to the "Linked"-part of Linked Open Data:

4. **Include links to other URIs, so that they can discover more things**

   This can mean using equivalence links, like owl:sameAs, to express that one of the resources you're expressing information about is the same as a resource somewhere else, e.g.

   ```
   ex:employee23765 ex:skill ex:interaction_design .
   ex:employee23765 owl:sameAs foaf:Kari_Normann .
   ```

   Other potential equivalence identifiers are skos:exactMatch, owl:equivalentProperty or owl:equivalentClass. This is an easy way of adding value to the Semantic Web and Linked Open Data cloud.

The Linked Open Data cloud has popularly been illustrated in a cloud diagram (see figure 2.1), where the size of the cirles correlate to the amount of links it has to other datasets, and the arrows indicating that a link exists between the two datasets.

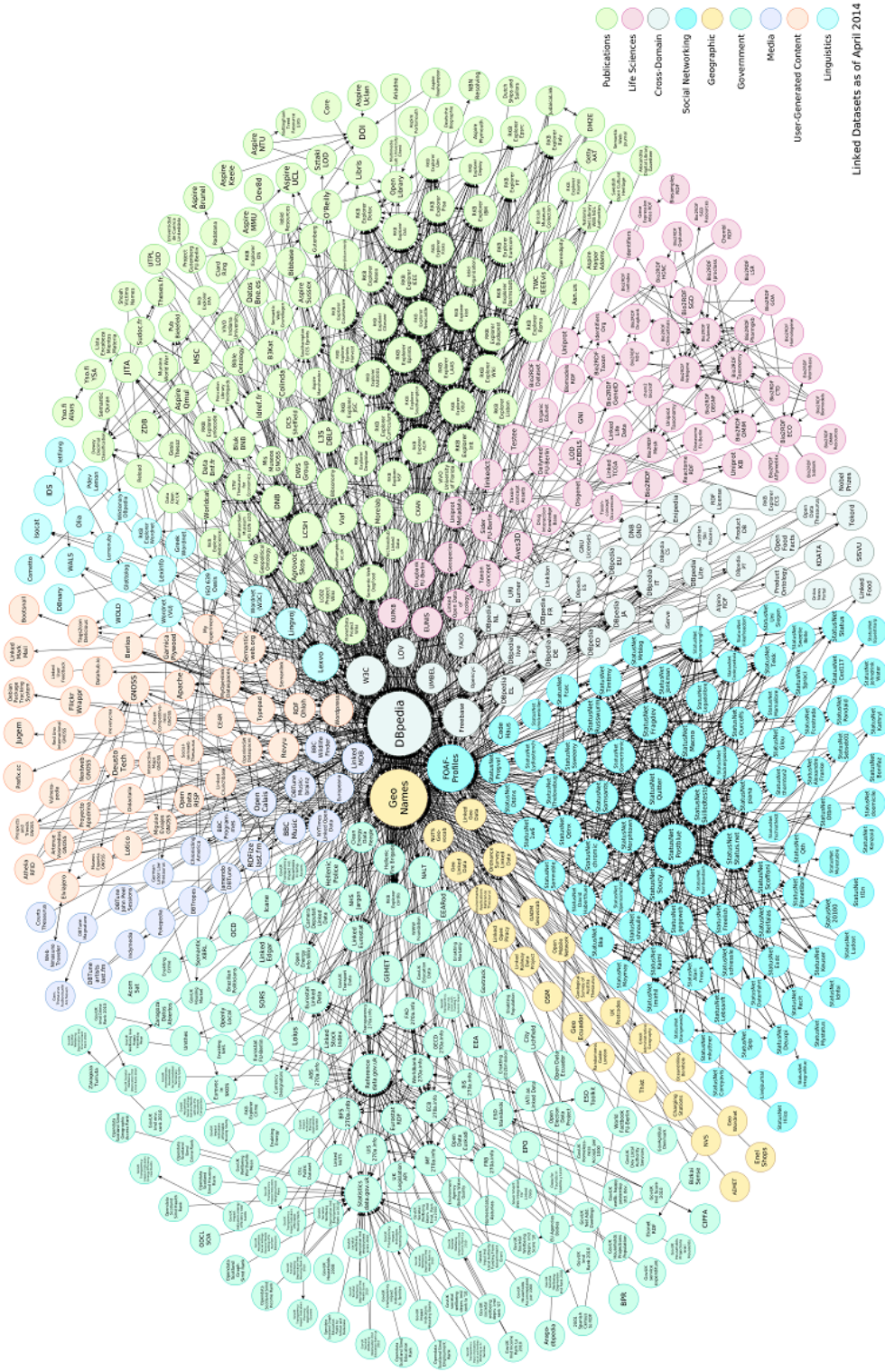
There are two primary types of Linked Open Data sources on the web today. One is through files, e.g. embedded in the text and metadata of the page. There are also small vocabularies and datasets available as files on the web, typically in one of the serialization formats like Turtle or RDF/XML. Another source of Linked Open Data is "behind" SPARQL endpoints. Endpoints are web addresses that users can send queries to, enabling them to access triple stores, also called knowledge bases. Some have an HTML-presentation with a webform, where you can type your query (either in SPARQL or another query language) in the text form. This is a very user-friendly approach for those who aren't accessing the endpoint via a piece of programming code or a script. Otherwise the query is sent as part of the URL. The endpoint then returns the results in a machine-friendly format, e.g. JSON. In the case of a webform, the endpoint displays the results on the page in HTML unless you choose otherwise. This is a form of content negotiation, which means having data available

[12] http://www.w3.org/TR/cooluris/

[13] http://www.ietf.org/rfc/rfc3986.txt

[14] http://data.dws.informatik.uni-mannheim.de/lodcloud/2014/.

Figure 2.1: The Linked Open Data cloud



The Linked Open Data cloud illustrated as a cloud diagram[14]. Each circle represents a knowledge base or dataset, and the arrows indicate links to other datasets.

in different formats at the same URI. W3 keeps a record of all available SPARQL endpoints[15]. One of these endpoints belong to DBpedia.

2.3.1 A crowdsourcing and Linked Open Data example: DBpedia

DBpedia[16] is a knowledge base based on semantically encoded data from Wikipedia, and has been one of the central projects in the Linked Open Data movement. Started by the Freie University of Berlin and the University of Leipzig, the English version of DBpedia currently describes 4 million things, whereas 3.2 million are placed in ontologies. It consists of roughly 800 000 persons, 600 000 places, 300 000 creative works, 200 000 organizations, 200 000 species and 5000 diseases.

DBpedia is considered by many to be a central part of the Linked Open Data movement, and is in the middle of the cloud diagram in figure 2.1.

Technically, DBpedia is a Virtuoso triplestore, which will by default display an HTML-page when accessed through a web browser. E.g. Per-Willy Amundsen's DBpedia page in figure 2.2. The Property-column lists all the types of relations that Amundsen has, and the Value column displays the values, e.g. his date of birth. The highlighted values have their own DBpedia-page, and the plain text values (e.g. the dbpprop:spouse Gry Anette Rekanes Amundsen) do not.

To extract triples from DBpedia, one can either follow the links on the pages, or query the SPARQL endpoint. The endpoint is also available as a HTML-page[17] and can return the information in multiple different formats (JSON, RDF/XML, XML, CSV etc.).

DBpedia as a source of information Since Wikipedia is DBpedia's main source of content, and Wikipedia is crowdsourced, it is natural to ask whether Wikipedia can be considered a reliable source of information. There has been published multiple research articles evaluating the quality of the information on Wikipedia, and the results have been quite satisfactory. It is even considered one of the most successful examples of peer collaboration. In spite of its large amount of contributors, studies have shown that incorrect information is corrected quickly, and that the quality of the content is as high as in traditional encyclopedias (Kittur and Kraut 2008).

If we were to use DBpedia, the semantic equivalent of Wikipedia, to extract semantic data, we should furthermore evaluate whether DBpedia is a reliable source of information. Zaveri et al. did a study on the quality of datasets on DBpedia, using both manual and semi-automatic processes, and identified four particular problem areas for the data quality on DBpedia (Zaveri et al. 2013). The first is the accuracy of the data. They found multiple instances where the triple or datatype was incorrectly extracted, resulting in inaccuracies.

[15] http://www.w3.org/wiki/SparqlEndpoints
[16] http://dbpedia.org
[17] http://www.dbpedia.org/sparql
[18] http://dbpedia.org/page/Per-Willy_Amundsen, viewed 4 May 2014

Figure 2.2: Per-Willy Amundsen in DBpedia

About: Per-Willy Amundsen

An Entity of Type : office holder, from Named Graph : http://dbpedia.org, within Data Space : dbpedia.org

Per-Willy Trudvang Amundsen (born 21 January 1971) is a Norwegian politician and Member of Parliament for the Progress Party. He is

Property	Value
dbpedia-owl:abstract	Per-Willy Trudvang Amundsen (born 21 January 1971) is a Norwegian politician and
dbpedia-owl:activeYearsEndDate	2013-09-30 (xsd:date)
dbpedia-owl:activeYearsStartDate	2005-09-12 (xsd:date)
dbpedia-owl:almaMater	dbpedia:Norwegian_School_of_Economics dbpedia:Sør-Trøndelag_University_College
dbpedia-owl:birthDate	1971-01-21 (xsd:date)
dbpedia-owl:birthPlace	dbpedia:Norway dbpedia:Harstad
dbpedia-owl:birthYear	1971-01-01 (xsd:date)
dbpedia-owl:nationality	dbpedia:Norwegians
dbpedia-owl:occupation	dbpedia:Politician
dbpedia-owl:party	dbpedia:Progress_Party_(Norway)
dbpedia-owl:profession	dbpedia:Economics
dbpedia-owl:region	dbpedia:Troms
dbpedia-owl:termPeriod	dbpedia:Per-Willy_Amundsen__1
dbpedia-owl:thumbnail	http://commons.wikimedia.org/wiki/Special:FilePath/PWAmundsen5789_2E_jpg_DF0
dbpedia-owl:wikiPageID	14175407 (xsd:integer)
dbpedia-owl:wikiPageRevisionID	590676697 (xsd:integer)
dbpprop:almaMater	dbpedia:Norwegian_School_of_Economics dbpedia:Sør-Trøndelag_University_College
dbpprop:birthDate	1971-01-21 (xsd:date)
dbpprop:birthPlace	Harstad, Norway
dbpprop:constituencyMp	dbpedia:Troms
dbpprop:dateOfBirth	1971-01-21 (xsd:date)
dbpprop:hasPhotoCollection	http://wifo5-03.informatik.uni-mannheim.de/flickrwrappr/photos/Per-Willy_Amundsen
dbpprop:imagesize	210 (xsd:integer)
dbpprop:name	Per-Willy Amundsen Amundsen, Per-Willy
dbpprop:nationality	Norwegian
dbpprop:occupation	dbpedia:Politician
dbpprop:party	dbpedia:Progress_Party_(Norway)
dbpprop:placeOfBirth	dbpedia:Norway dbpedia:Harstad
dbpprop:profession	dbpedia:Economics
dbpprop:shortDescription	Norwegian politician
dbpprop:spouse	Gry-Anette Rekanes-Amundsen
dbpprop:termEnd	2013-09-30 (xsd:date)
dbpprop:termStart	2005-09-12 (xsd:date)
dc:description	Norwegian politician
dcterms:subject	category:1971_births category:Living_people category:Members_of_the_Parliament_of_Norway category:Norwegian_politicians category:Norwegian_state_secretaries category:People_from_Harstad category:Progress_Party_(Norway)_politicians

An example of a DBpedia page[18]. This is the HTML-representation of the entity Per-Willy Amundsen in the knowledge base DBpedia. The headline ("Per-Willy Amundsen") is the subject of each triple, and the left column show the various predicates, meaning the relationship the entity or subject has. The right column display the objects in the triples, which are either resources (other entities in DBpedia, e.g. dbpedia:Troms), or literals ("Harstad, Norway").

Another issue was data relevancy. Some of the data extracted from Wikipedia was not relevant for DBpedia users, like information on images that were only available on the corresponding Wikipedia page.

A third issue is representational consistency, and was most prevalent in number extraction. This was often caused by an inconsistency on Wikipedia is how a particular number was written (e.g. 20.000 instead of 20 000).

Finally, interlinking turned out to be a problem because many Wikipedia pages contains link to either external web pages or is interlinked with other datasets elsewhere. Some of these links are either dead, or don't contain useful information.

In spite of these issues, the authors judge DBpedia to be a reliable source of information about areas like the media, e.g. movies and actors. However, it's still not suitable for more complex uses e.g. as a medical database.

2.3.2 Other large knowledge bases

Wikidata Wikidata[19] is another very large knowledge base that describes many of the same things as DBpedia. But while the goal of DBpedia is to create a knowledge graph from Wikipedia, Wikidata aims to offer a knowledge base that anyone can edit. They do not extract any knowledge from Wikipedia, although it does contain data from DBpedia as well. The data is accessible through data dumps or various APIs and endpoints[20]. Another large knowledge base, Freebase[21], is soon to be merged with Wikidata, adding even more triples.[22]

LinkedMDB LinkedMDB[23] publish movie-related information as Linked Open Data, and has currently published roughly six million triples. These contain more than 500 000 links to other movie pages, and more than 120 000 links to other knowledge bases in the Linked Open Data cloud.

YAGO Another large knowledge base is YAGO[24], by Max Planck Institute for Computer Science in Germany. YAGO contains triples from GeoNames, WordNet[25] (which is a lexical database), and ten Wikipedias in different languages.

[19] http://www.wikidata.org

[20] http://www.wikidata.org/wiki/Wikidata:Data_access

[21] http://www.freebase.com

[22] https://plus.google.com/109936836907132434202/posts/3aYFVNf92A1, viewed 4 May 2015

[23] http://www.linkedmdb.org/

[24] http://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago//

[25] https://wordnet.princeton.edu/

GeoNames GeoNames[26] focuses on geographical locations, and has more than 10 million names of locations altogether. 600 000 of these are in Norway, making Norway the second most mentioned resource in the database, just below the United States. The data is available through the downloadable data dumps and numerous web services.

2.4 Examples of use

2.4.1 BBC

Historically BBC have focused on maintaining multiple subsites (food, music etc.), each publishing a large amount of audio, video, and text content. Although there are great user experience and navigational opportunities for a site of this size, some of these have been missed due to lack of data interlinking. In collaboration with Freie Universität Berlin and Rattle Research, BBC set out to accomplish the following goals: (1) to link BBC's content to the Linked Open Data cloud, making traversing the graph easier for both users and developers, (2) use existing identifiers to classify their content, in addition to (3) developing their own identifiers (Kobilarov et al. 2009).

Demonstrating use of Linked Open Data in BBC Programmes

BBC Programmes was considered a good place to start, because although the BBC broadcasts 1000 to 1500 programmes a day, not all programmes had their own page. Furthermore, there were great variations in how much content each program page had; some had detailed descriptions with lists of the crew and cast, while others only displayed upcoming broadcasting dates (Raimond et al. 2010). Starting in 2007, the idea was to use DBpedia to serve as a common vocabulary and suggest tags. *Tags* are commonly used to express certain characteristics of different types of content, and for BBC indicated what the content item, in this case a program page, was about. Tags are sometimes used in addition to categorization, as the category doesn't necessarily express enough information in itself.

They began by assigning web identifiers to all BBC programmes (TV-series, episodes etc.), before linking them to DBpedia using owl:sameAs. This way they became part of the Linked Open Data cloud, while simultaneously being able to utilize it. Each web identifier had content negotiated representations in JSON, XML and RDF/XML, which are all machine-readable languages. They proceeded to make "about"- and "features"-links to people, places and subjects, e.g. programmes:segment <features> music:track. Instead of using the program names they assigned web identifiers like http://www.bbc.co.uk/programmes/b00c6dv5. This ensures that a link doesn't become outdated or broken if the program changes names, and doesn't clash if another program shares the same name.

[26] http://www.geonames.org

Interlinking with BBC Music

BBC Programmes is closely linked to BBC Music, as programmes (radio or television) often feature music artists, album reviews and tracks. Linking the new BBC Program pages to a new BBC Music (Beta) demonstrates the cross-linking between subsites which is enabled by Linked Open Data. They started by making unique web identifiers to the different music objects: artists, genres, releases and their reviews. All the information displayed on the new music pages were from MusicBrainz, which provides information on the artists' releases and external pages, DBpedia which provides background info on artists, and BBC who provides additional content like audio snippets and images. Importing Linked Open Data from other knowledge bases like DBpedia and MusicBrainz was a way to *semantically enrich* the site, without having to provide the content themselves. Linking BBC Programmes to BBC Music provided new opportunities regarding functionality and user experience on their site. One example is artist recommendations. Typically when a user is recommended similar artists to an artist they like, he or she is presented with a variety of suggestions, but with no information on how they are alike or what the recommendation algorithm is based on. Using linked data the path has a name, this can be displayed to the user.

The interlinking process

BBC was already using an auto-categorization system called CIS, with the top categories being Subject, Brand, Time Period, Place and Proper Name, in addition to a more general vocabulary and a list of locations. The role of CIS was to categorize BBC Program pages automatically based on their textual description, which would create a link to other programs or news stories with the same tag or category. But the tags were not linked to other tags, e.g. an article with tagged with "Beijing" would not be related to an article with a "Beijing Olympics"-tag. BBC wanted a richer mapping, with related and equivalent terms, and DBpedia was their solution.

The first step was linking CIS concepts to DBpedia URIs, which was done by building an algorithm that matched the name of the category to DBpedia-pages. Whenever DBpedia would return multiple page alternatives, they would rely on contextual information to denote which would be the best match, like words in parentheses (e.g. "Mary (1985 sitcom)") and other concepts in the same category. Another way of identifying the correct URI was to do a weighted label lookup. This method bears similarity to PageRank, and works by counting the amount of Wikipedia inter-article-links that points to it. This gave an additional indication of which DBpedia page was the most relevant.

Moving on to categorizing documents, as opposed to concepts and other structured data, they developed a named entity extraction system called Muddy Boots. Using Named Entity Recognition (NER) combined with the Yahoo Term Extraction API, Muddy Boots' primary objective was to identify the main entities of any document, and enable BBC to

use DBpedia as a controlled vocabulary. Applying a similar technique as with the concepts, it firstly matched an entity with the name of a DBpedia resource. It then used the complete list of extracted terms from the rest of the document as contextual information, which was used to rank the returned DBpedia page titles.

Additionally, BBC had to develop a tool that could manually add or remove DBpedia links from BBC documents, which was added into the graphical user interface (GUI). Changing a DBpedia-link would immediately change the links to related articles on the page, to make the concept of linking more tangible and thus more interesting to apply for the users/journalists.

BBC published their Programmes ontology in November 2007[27], and have since published multiple other ontologies[28]. As a result of the new controlled vocabulary, BBC was able to generate *topic pages*. These are pages that contain news content, which is unstructured, together with structured BBC Programmes content. Not only do pages like this focus search engines, but they also provide readers with a bridge between subsites. DBpedia serves as a vocabulary for the topic pages as well, so it can provide even more contextual info from each DBpedia-page, like geolocation, place of birth, place of death etc..

2.4.2 The New York Times

Like BBC, the NY Times has enormous amounts of content. In 1913 they published the first issue of The Times Index, which contained a cross-referenced guide to all the names, articles and items appearing the past three months[29]. This practice of publishing subject headings continued yearly until modern databases became the norm.

The NY Times thesaurus consisted of five different controlled vocabularies: personal names, organizations, subjects, geographical locations and titles of various types of creative work. The disadvantage of simply tagging content without adding much structure was similar to that of the BBC, which is lack of interlinking. They could provide the user with all the articles written on a given person, but not his date of birth. At the 2009 Semantic Technology Conference they announced the release a NY Times Thesaurus as Linked Open Data, as part of their TimesOpen strategy. Their aim was to map the approximately 30 000 tags behind their Topic pages[30].

Another goal was aiding third parties in accessing their content more easily through their open API, which could help to spread their content to other users and increase traffic to NY Times.

The NY Times chose a similar, albeit more strenuous approach than the BBC. After consulting experts in the Semantic Web community, they manually mapped more than 5000 person name subject headings to DBpedia and

[27] http://www.bbc.co.uk/ontologies/po

[28] http://www.bbc.co.uk/ontologies

[29] http://www.nytimes.com/2001/11/17/opinion/dusting-off-the-search-engine.html

[30] http://open.blogs.nytimes.com/2009/06/26/nyt-to-release-thesaurus-and-enter-linked-data-cloud

Freebase[31]. Each name was given a URI containing a long sequence of numbers, e.g. Joe Biden is http://data.nytimes.com/N5760378394067866992. These have been published as Linked Open Data under the Creative Commons 3.0 Attribution Licence, and they've even launched The New York Times Linked Open Data Community. In January 2010 they announced the mapping of approximately 5000 more new subject headings, this time focused on organizations, publicly traded companies and geographic identifiers[32]. GeoNames were used for the geographic identifiers.

Throughout this project NY Times have been consistently encouraging the public to use the data through their API, even publishing a blog entry on how to build your own NY Times Linked Data application[33]. They also host TimesOpen events yearly, in addition to hackatons [34].

2.4.3 Detecting trending topics in German news agency

In *Towards Topics-based, Semantics-assisted News Search* published in 2013 Martin Voigt, Michael Aleythe and Peter Wehner set out to develop a tool that would automatically identify upcoming and current topics in a stream of news articles. The goal was to present these in an ordered list to the end user, and in this way provide journalists and other news agency employees with valuable information on current topic trends (Voigt, Aleythe and Wehner 2013).

They identified four phases that each news article would enter. The first phase was *Pre-processing*, which entails extracting semantic data from each news item. As this was a German news agency, they first had to determine whether the text was in English or in German. By checking for common English words like "of" and "for", the language was detected with 99 percent precision. The next step in this phase was categorizing the article, which was done with a tool called LingPipe[35]. LingPipe is a text processing tool with a Java API and support for multiple languages. They used its NaiveBayesClassifier class, which is a probabilistic classifier, along with the IndoEuropeanTokenizer. The tokenizer splits a string into different parts, usually words, in order to do further calculations on each item. The tokens produced by the tokenizer is used to determines which category the article most likely belongs to, e.g. Sports or Entertainment.

To extract knowledge from the article after categorization, Voigt et al. employed two Named Entity Recognition (NER) techniques. The wordlist-based NER identified terms using wordlists from Freebase, the German DBpedia[36], GeoNames, and YAGO. Next they applied statistical NER to

[31]http://open.blogs.nytimes.com/2009/10/29/first-5000-tags-released-to-the-linked-data-cloud/, viewed 4 May 2015

[32]http://open.blogs.nytimes.com/2010/01/13/more-tags-released-to-the-linked-data-cloud/, viewed 4 May 2015

[33]http://open.blogs.nytimes.com/2010/03/30/build-your-own-nyt-linked-data-application/, viewed 4 May 2015

[34]http://open.blogs.nytimes.com/2009/02/26/open-doors-open-minds/, viewed 4 May 2015

[35]http://alias-i.com/lingpipe/

[36]http://nl.dbpedia.org

identify named entities not appearing in the word list. For the English articles the Stanford Natural Language Processing Tools[37] was used, but they had trouble finding a good tool for the German articles. The authors recommend relying on the wordlists for the German words, although these often don't include local persons and organizations.

The following phase was *Data Storage*, in which the semantic data extracted in the previous phase is saved in a knowledge base, and the article itself in a relational database. After creating knowledge base benchmarks for their unique case they decided on the Oracle 11gR2 which allows for combining their relational database and knowledge base. This solution was tested using data from the Main-Post, a German news agency, which was continually imported to simulate the growth and amount of data it will have to handle.

In the *Post-processing* stage they identified topics by recognizing two or more items frequently appearing together. Their importance depended on how many times they appeared in a day, and the time period specified determined whether it was trending. The first step in this process was organizing every article as columns and rows in a triangular matrix. The similarity between each of the articles was determined using the Dice coefficient and the named entities extracted earlier. As some articles have duplicates, the articles with a similarity of 1.0 (meaning they are identical) were merged, as well as other very similar articles. Using the Complete Linkage-method, a hierarchical type of clustering, the similarities were computed once again and the new values entered into the matrix. These steps were repeated until the similarity values reached zero, i.e. they continually merged articles into the same rows and columns. Next they removed topics with very few articles, and topics with very many, as these topics are respectively considered too narrow or too broad. Finally the topics are stored within the knowledge base, and linked to specific dates. The "topic model" is only valid for one day, to reflect the changing and ever-evolving nature of news content.

Voigt et al. also connected the geographical names to the corresponding triples in GeoNames, which allowed them to do spatial clustering, i.e. give editors a view of news within a particular region.

Lastly they developed a search component to gain access into the new data collected. An index for every author, agency, headline and topic was built, which could be searched using simple keywords or faceted search. The web interface provided a view for the topics, the articles, and related articles as a similarity value was calculated in the post-processing stage. Another idea was to display the current trending topics on big screens on the walls of the newsroom, giving journalists and editors a quick view of the data.

[37] http://nlp.stanford.edu/software/

2.4.4 The Guardian

> "Implementing a Linked Data approach across our content should lead to better tools for journalists, better services to sell to our business partners, and, ultimately, better story-telling with which to reach and inform our audiences"

— Martin Belam, former Lead User Experience and Information Architect at the Guardian.[38]

The Guardian is another large British digital news agency that has mapped parts of its content to the Linked Open Data Cloud. Already in January 2010 they organized a News Linked Open Data Summit together with BBC and Media Standards Trust discussing the opportunities Linked Data could offer the news industry. The trigger seemed to be the success of the BBC Wildlife Finder (another BBC project utilizing Linked Open Data, outlined in Raimond et al. 2010) and especially the UK government's plan to release large datasets as Linked Open Data. In his blog post, Martin Belam, former Lead User Experience and Information Architect at the Guardian, visualizes a future where public entities, e.g. schools, have unique IDs in a large knowledge base published and maintained by the government, and that all information published from various sources on that particular entity contains a link to the corresponding ID. This kind of interlinking would greatly enhance journalists' ability to extract valuable data in the case of an event regarding that particular entity. The key, he writes, is collaboration. Not in terms of a single ontology, but technical standards, and making them interoperable.

In October 2010, The Guardian posted a blog post [39] on their efforts to map every tag and article about books to their respective ISBNs, and every artist and band to a MusicBrainz ID. MusicBrainz [40] is a large knowledge base that provide information on almost one million artists and bands, and 18 million tracks [41].

In 2010 The Guardian had already had their content available to the public through their Content API [42], where roughly 1.2 million pieces of content was available at the time [43]. Prior to the mapping, The Guardian already had tools for adding external identifiers to tags and content items, which were used to pull information from other sources to their sports pages etc.. This was called a "reference" field, which was a multivalued string field.

The same functionality was used for adding ISBNs and MusicBrainz IDs, but the reference field was now exposed to outside parties. Each Linked Open Data reference was represented as <type>/<value> in the

[38] http://www.theguardian.com/help/insideguardian/2010/jan/25/news-linked-data-summit, viewed 4 May 2015

[39] http://www.theguardian.com/open-platform/blog/linked-data-open-platform, viewed 4 May 2015

[40] http://musicbrainz.org

[41] http://musicbrainz.org/statistics, viewed 2 May 2015

[42] http://open-platform.theguardian.com/

[43] https://youtu.be/greXtGJjtIg

reference field, e.g. isbn/9781847249746. In October 2010 about 600 artists and bands had been mapped, and in August 2011 they published around 3 million album pages [44]. These pages were generated automatically, combining content from their Content API, LastFM and Amazon among others [45], and all contained a disclaimed informing the users that the page was "automatically assembled and may not be entirely accurate", along with contact information encouraging the user to report any parsing errors. Their view was, as Belam writes, that they "would rather have the 3 million pages live with the opportunity to correct mistakes, than spend the time and money auditing them in advance."

Adding these external references (ISBNs and MusicBrainzIDs) not only produced a huge amount of album pages, but also aid users in finding content from The Guardian on the given entity. The content extracted from the Guardian could be combined with the user's own content or other data available in the Linked Open Data cloud, e.g. abstracts from DBpedia/Wikipedia.

Each journalist is now encouraged to add the ISBN to infoboxes on books, and in 2010 about 2800 ISBNs were mapped to various content items.

This chapter has outlined some Semantic Web standards and concepts, in addition to explaining crowdsourcing, Linked Open Data, and some of the previous uses of Linked Open Data in the news publishing industry.

From the above examples of use, one can conclude that Linked Open Data has been used in a multitude of ways. BBC, NY Times and The Guardian wanted to facilitate third-parties use of their content, which was done through mapping parts of it to Linked Open Data identifiers. These mappings also enabled them to improve their pages through better interlinking of content, and semantically enrich it by importing data from existing Linked Open Data knowledge bases. The German news agency used Linked Open Data as a vocabulary or word list in identifying trending topics, and also used the information in the GeoNames knowledge base to make a map showing news in each region. Furthermore, BBC used a Linked Open Data knowledge base, DBpedia, as a controlled vocabulary, essentially "outsourcing" it.

[44] http://www.currybet.net/cbet_blog/2011/08/guardian-album-pages.php, viewed 4 May 2015

[45] http://www.theguardian.com/info/developer-blog/2011/aug/02/music-album-pages, viewed 4 May 2015

Chapter 3

Suggestions for use of Linked Open Data

This chapter presents my suggestions for use of Linked Open Data for VG. Some of these are ideas from previously mentioned examples of use, like outsourcing the controlled vocabulary as BBC did. The BBC also generated richer topic pages through interlinking enabled by Linked Open Data, which is the subject in section 3.3. Furthermore, they've demonstrated *semantic enrichment*, a concept which is further explained in section 3.5. Other suggestions surfaced through collaboration with the development department in VG, like the fact-checking tool. The rest of the suggestions, like using knowledge extraction tools for tag suggestions, and reasoning to produce data, are inspired by the other ideas.

Some of the suggestions has thus been developed previously by other parties, but as this chapter will show, the approaches used there aren't necessarily as suitable for VG. Each following section outlines an idea, followed by potential challenges.

3.1 Use Linked Open Data knowledge extraction tools for tag suggestions

Tagging and categorizing is essential when dealing with large amounts of content. In a news publisher's case, the *category* will usually express the scope or another top-level property of the article. E.g. if it's an article on a local football team, it will be in the Sports category, and might be placed in a more specific category within Sports, like Football. *Tags* are usually more specific, but can belong to several domains. This is one of the key advantages of tagging, which is the ability to express meaning that can reach across multiple categories, and be as specific or unspecific as you'd like. A tag could refer to a name, an event, an organization etc. The practice of categorizing and tagging support many of the ways news publishers convey their content to the readers, like organizing the different types of news in the main menu (regional news, international news etc),

and providing related articles as contextual navigation.

In 2014, Schibsted Media Group, the owners of VG, announced an increased focus on personalization, which means increasing the focus on customization to each unique user to improve the user experience [1]. Schibsted aims to achieve this through analyzing the data that is collected on each user, a trend which has become more and more prevalent with services like Netflix and Amazon. These are services that make recommendations based on users' previous actions to provide them with products that they are more likely to respond to, and this way increase revenue. In practice, focusing on personalization means discovering what types of content each user consume, and provide them with similar content. One of the main conditions for achieving this is being able to tell what the user is actually consuming, and this is where tagging and classifying content becomes important. In this context tagging and categorizing go beyond simply presenting content in a clear and logical way, but can also be used in far more advanced algorithms used for personalizing the user experience.

VG has multiple different subsites, like VG+, VGTV, E24 etc., and all of these have their own category tree. A part of vg.no's category tree is visible from the front page, e.g. the top categories Regional (*Innenriks*), International (*Utenriks*), Sports etc.. In addition to being categorized, each content item is tagged with zero to three tags. The first tag is the *primary tag*, and should be the most specific one. The next tags are *secondary tags* and are more general. Although this is effective in theory, it's not necessarily as straight forward in practice. There are many pitfalls, like journalists only using the tags they are familiar with, forgetting to tag, inconsistent tag use, ambiguous tag names, multiple tags for the same concept, not enough tags available, or too many tags available.

Linked Open Data in not just data on entities in large knowledge bases, but can also be thought of as a vocabulary (outlined in section 2.1.2) due to its structure. Since every entity belongs to a class, and these classes have superclasses and subclasses, knowledge bases include names on both the entities themselves, the categories they belong to, and what categories the categories belong to etc. Ultimately the knowledge bases store a large number of names. These names could be used to aid tagging by providing journalists writing articles with tag suggestions from the existing tag database. One way of doing this is with the use of knowledge extraction tools. *Knowledge extraction* in this case means identifying entities within a text, like the entities "Hillary Clinton" and "Germany" in the text "Hillary Clinton visited Germany last week". There are many knowledge extraction tools available, and some of them use Linked Open Data knowledge bases as vocabularies. Given that an entity has a corresponding URI in the knowledge base, the software will return the URI for each entity it finds. E.g. if the knowledge extractor recognizes "Hillary Clinton" as an entity in a text, and uses DBpedia as a knowledge base, it would return

[1] http://www.schibsted.com/en/Press-Room/News-archive/2014/Data-and-data-analytics-provide-better-services-/, viewed 4 May 2015

`http://dbpedia.org/resource/Hillary_Clinton`

. This URI can be used to find labels belonging to the entity, and match these labels to the tags in the existing database that VG maintains. This way knowledge extractions tools using Linked Open Data could be used to suggest proper tags from the existing tag database to improve tagging quality. This functionality could be implemented as a separate plug-in in their publishing system, or within the existing plug-in used for tagging called "Metadata". This is a simple way of using Linked Open Data without having to make any big changes - it simply supports the structures that are already there.

3.1.1 Knowledge extraction software

There are multiple different knowledge tools able to return Linked Open Data identifiers. Among the most popular is DBpedia Spotlight[2], a service that annotates text documents with DBpedia URIs. It's available as a web service and for downloading locally, and is published under an open source licence. The web service demo [3] demonstrates how it works by taking text input, processing the text using natural language processing techniques, and returning the text with annotations. The DBpedia entities identified in the text are highlighted, and contain the links to the corresponding DBpedia URIs. Pasting the links into braces, it can look something like this [4]:

```
After years of stagnation, there's been a burst of activity in
transferring inmates out of the controversial Guantanamo Bay detention
camp [http://dbpedia.org/resource/Guantanamo_Bay_detention_camp].
Where will it end?

Five detainees - all Yemenis - are leaving Guantanamo, four to go
to Oman [http://dbpedia.org/resource/Oman] and one to Estonia
[http://dbpedia.org/resource/Estonia]. They are the latest in a
flurry of transfers out of the prison, part of a new effort to
close the facility down.

Over the past year, 28 detainees have been transferred out of the
prison and taken to countries such as Kazakhstan
[http://dbpedia.org/resource/Kazakhstan] and Uruguay
[http://dbpedia.org/resource/Uruguay].

This leaves 122 men who are still held at Guantanamo. One of
them is Shaker Aamer [http://dbpedia.org/resource/Shaker_Aamer],
the last British [http://dbpedia.org/resource/United_Kingdom]
resident being held in Guantanamo Bay
```

[2]https://github.com/dbpedia-spotlight/dbpedia-spotlight/wiki
[3]http://dbpedia-spotlight.github.io/demo/
[4]http://www.bbc.com/news/world-us-canada-30820897, viewed 2 May 2015

```
[http://dbpedia.org/resource/Guantanamo_Bay_detention_camp].

Mr Aamer, who is from London [http://dbpedia.org/resource/London],
has been held at Guantanamo since 2002.

Prime Minister [http://dbpedia.org/resource/Prime_Minister_of_the_
United_Kingdom] David Cameron [http://dbpedia.org/resource/David_Cameron]
is visiting Washington this week and is planning to ask US President
Barack Obama [http://dbpedia.org/resource/Barack_Obama] about
Mr Aamer's release.
```

DBpedia Spotlight also has functionality for filtering the results by entity types, e.g. people or events, and receiving the result in a different format.

3.1.2 Challenges

In using any of the knowledge extractors mentioned above there's a high risk of both false positives and false negatives. There is seemingly a lot less data on Norwegian entities available, meaning that many of the services will not be able to recognize important Norwegian people as Linked Open Data. This can result in false negatives.

Another challenge is not about the content itself, but its Norwegian syntax. In using tools made for the English language there's a risk of it identifying entities where there are none. A quick test in the web service demo revealed that the possibility of false positives might be very high, as in this example:

```
I finaleserien står det 2-2 i kamper mellom Stavanger Oilers
[http://dbpedia.org/resource/Stavanger_Oilers] og Storhamar
Dragons [http://dbpedia.org/resource/Storhamar_Dragons]. I kveld
er [http://dbpedia.org/resource/ER_(TV_series)] det ny match
i best av syv. Vinner laget til Thoresen har dem fordel
[http://dbpedia.org/resource/Fordell_Castle] torsdag - med ny
hjemmekamp - i elegante DNB Arena
[http://dbpedia.org/resource/DNB_Arena_(Stavanger)]
i Stavanger [http://dbpedia.org/resource/Stavanger].

- Jeg er [http://dbpedia.org/resource/ER_(TV_series)] mer sammen
med bikkja enn kona, sier Petter Thoresen
[http://dbpedia.org/resource/Petter_Thoresen_(ice_hockey)].
```

The text in the example is from a VG article [5], and the annotated DBpedia-links are again in braces. In this example, DBpedia Spotlight

[5] http://www.vg.no/sport/ishockey/jeg-er-som-nils-arne-eggen-bare-ikke-saa-hoeyt-utdannet/a/23433955/, viewed 2 May 2015

annotated the words "er", meaning "is", with the URI for the 90s TV-series E.R. , and the word "fordel", meaning "advantage" as the 16th century tower house, Fordell Castle, in Scotland. A possible solution to this is adding stop words, which is common words that should not be included in the knowledge base search, but this is only possible when downloading the software.

3.2 Outsource the controlled vocabulary

A *controlled vocabulary*, as explained in chapter 2, is a set of standardized terms. In a news publisher's case, the controlled vocabulary refers to the names of categories and tags.

As mentioned earlier, each content item belongs to one category (which will usually belong to another category etc.) and have zero to three tags. In VG, the tags are created by around 30 journalists with access to the tag database. Previously, when all the journalists were able to add tags, the result was multiple tags referring to the same entity, with slight variations in spelling etc.. For this reason, the few journalists with access have been given special training to create them correctly.

According to the VG's Head of Software Engineering, Tommy Jocumsen, the tagging practices in VG are in need of improvement. There are inconsistencies in which tags are for articles belonging to the same story, which results in incomplete topic pages. Ultimately some articles become harder to find among all the other content. Some articles are simply tagged "Crime" (*Krim*) or "Hollywood", even when it refers to a fairly well-known case or person, because there simply doesn't exist a more specific tag. To reach Schibsted's goal of personalization, this is one of the areas that might need revision.

The previous section explained how Linked Open Data knowledge extractors could be used to facilitate correct tagging with the existing VG vocabulary. Another option is using a Linked Open Data vocabulary as tags, instead of the existing ones. Two of the most appropriate alternatives would most likely be either DBpedia or Wikidata, as both of these cover a wide range of topics and entities. This would be a way VG could *outsource* tag management to a certain extent, like the BBC did. In practice this would entail downloading the knowledge base locally, and the name of each tag would be one of the attributes of the entities, like the rdfs:label in DBpedia, e.g. in http://dbpedia.org/resource/Barack_Obama rdfs:label "Barack Obama". When a new tag should be created, it will be added to the knowledge base; either directly to the global one, or only to the local version. VG would have to update its local version with regular intervals to ensure that the vocabulary is up to date. This is also dependent on the global knowledge base containing updated information in the first place, which is something that has to be assessed beforehand.

Given that the Linked Open Data vocabulary is more specific than the current one, switching vocabularies could mean generating more specific topic pages, e.g. for a certain artist. It could also result in more general

topic pages, like "Asia", even when the tag is more specific, like "India", due the hierarchical relationship between the entities in the knowledge base, which would transfer to the tags in the vocabulary. VG could even keep their existing practice of tagging with zero to three tags, and could apply reasoning tools to decide which articles are about Asia. Another option is increasing the amount of tags possible on each article, and tag the content item with all the tags at once (both the highly relevant and the slightly less relevant ones), and only use the reasoning tools for special purposes. In using Linked Open Data as a vocabulary, the knowledge extraction tools mentioned in the previous section could be used to facilitate correct tagging.

3.2.1 Challenges

One of the main challenges is that DBpedia and other Linked Open Data knowledge bases don't necessarily contain URIs, and thus tag names, for popular Norwegian entities, as mentioned earlier. If VG were to use a knowledge base like DBpedia as a controlled vocabulary, they should be able to add new entities to the knowledge base, and in this way create a tag. This approach requires the journalist to be at least partly familiar with the structure of the knowledge base, although there is most likely ways of making it easier to understand, and develop tools to support it. Furthermore, a new tagging scheme would impose changes on people's working habits, for the journalists in particular. Not only would the tags be different, which might be difficult to adjust to, but the way of searching and browsing through tags might also change, which would require the journalists to change habits that for some have formed over the span of many years.

Another challenge is that in adopting a Linked Open Data vocabulary directly, VG might end up with an English controlled vocabulary. However, the Norwegian name for each entity is available through the owl:sameAs-relationship in DBpedia. This relationship almost always have a link to international versions of DBpedia, including a Norwegian one (which doesn't appear to be available, but include the name nevertheless). E.g. dbpedia:Germany owl:sameAs "http://no.dbpedia.org/resource/Tyskland".

3.3 Generate rich topic pages through enabling interlinking

VG, like many other news agencies, maintain multiple different subsites. Besides the main page vg.no, there's VG+, VGTV, E24, Vektklubb, MinMote, VG Live and many more. When a news article for vg.no is given a tag, this tag is displayed as a link in the article, see figure 3.1, which will send the user to a form of *topic page*. The topic page contains all the other articles from VG.no that are tagged with the same tag. The side bar displays the most recent articles on vg.no, recent VGTV-videos and picture galleries (*bildespesialer*), see figure 3.2. In other words, the only content available on

Figure 3.1: A VG tag example

The tags on an article are presented in the form of a link. The link leads to a topic page with all the other articles tagged with the same tag. [6]

the topic page is content from vg.no, which are articles presented in a list. Anything besides this is other general content, but not content specifically related to the topic of the topic page. Albeit informative, the topic pages in VG does not display *all* the content available on a topic, like VGTV videos, but only text articles specific to vg.no.

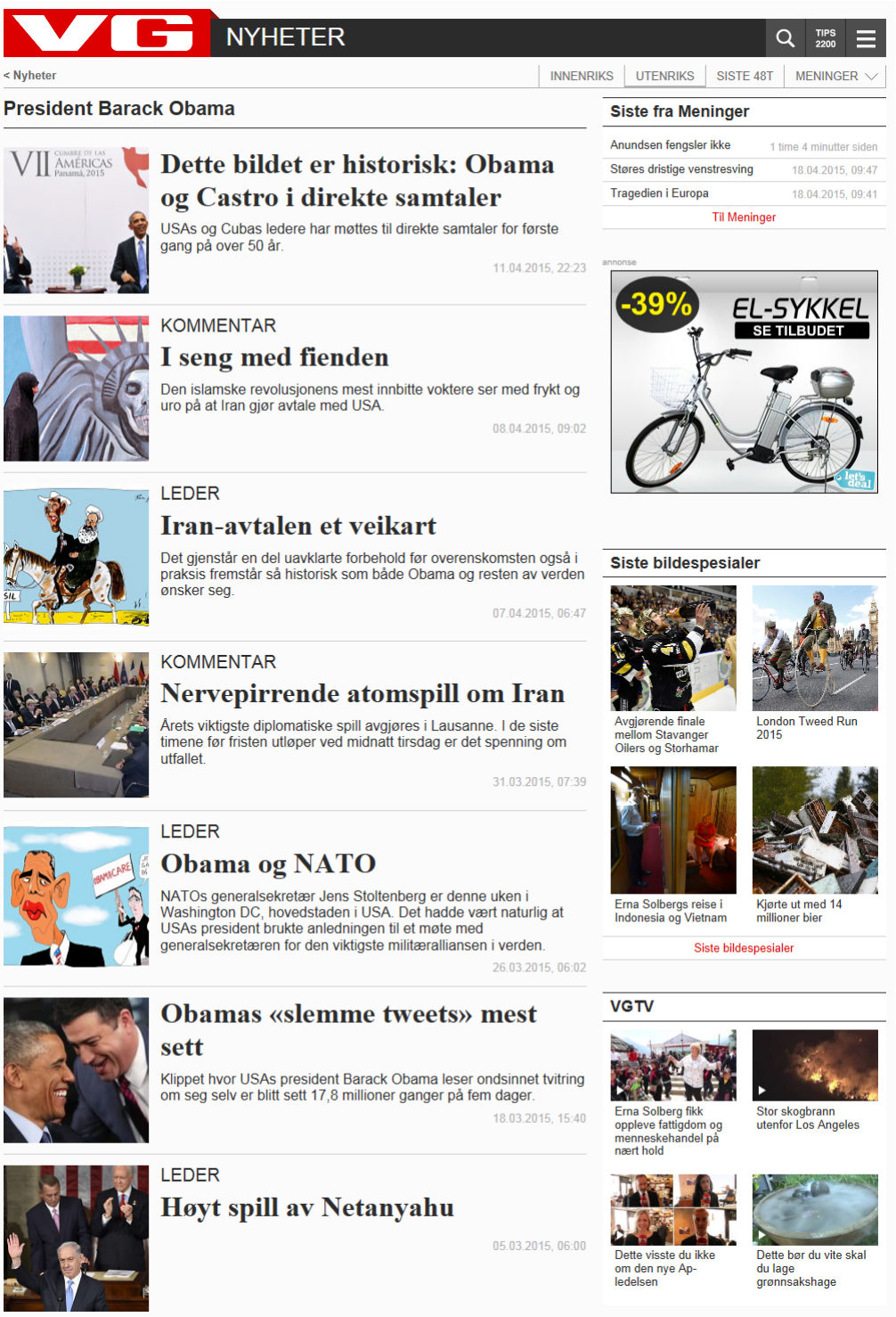
NY Times is an example of a news publisher with very rich topic pages. E.g. the topic page for Barack Obama [7] reveals multimedia content, "Highlights from the archives", related links and related articles, see figure 3.3. Additionally you can subscribe to e-mail alerts or an RSS-feed on the topic. The Guardian also combines various content types, see figure 3.4. This topic page on Rihanna contains news articles, blog posts, reviews, videos and picture galleries, organized by the date and/or month of the year. Another example is the BBC Music pages, who have a page on each music artist. This page on Ed Sheeran, see figure 3.5, which displays a main photo, information on when he was born, a link to his Wikipedia page, video clips, a section called "past BBC events" etc. Some of this information is imported automatically from outside sources, but a lot of the content is from various BBC subsites. Producing a topic page like this is possible through the interlinking between the subsites.

[6] http://www.vg.no/forbruker/reise/reiseliv/lover-ny-langrute-hvert-aar-si-hvor-du-vil-reise/a/23434915/, viewed 16 April 2015

[7] http://topics.nytimes.com/top/reference/timestopics/people/o/barack_-obama/index.html

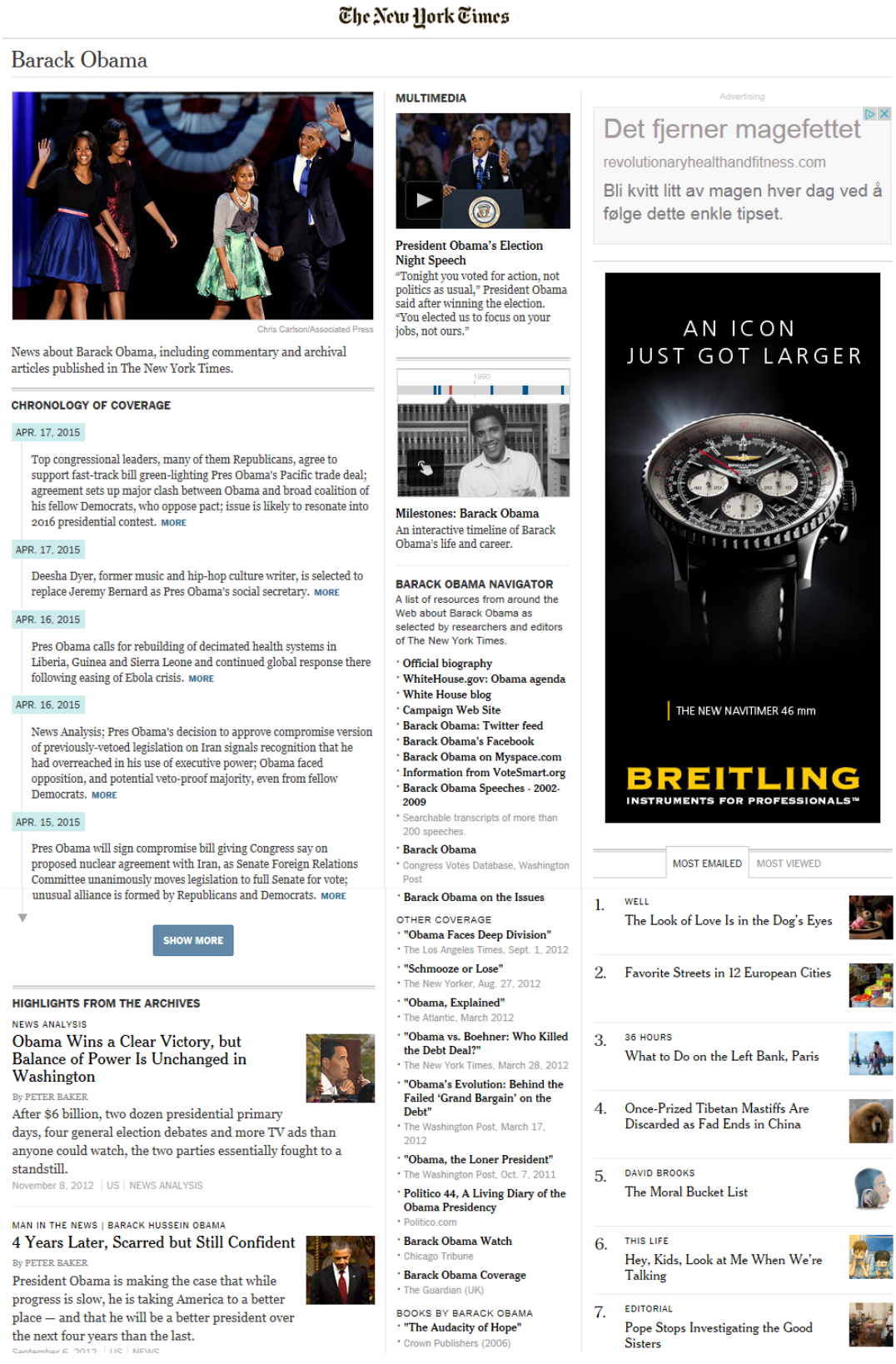
[8] http://www.vg.no/nyheter/utenriks/president-barack-obama/, viewed 8 April 2015

Figure 3.2: VGs current topic page on Barack Obama



The page[8]includes links to articles tagged with "Barack Obama", but doesn't contain any other content on the subject. The picture galleries and other multimedia content are simply the latest content available from each respective site

Figure 3.3: NY Times' topic page on Obama



The NY Times topic page on Barack Obama display a timeline of their coverage, "highlights from the archives", a "Barack Obama navigator" with multiple links, in addition to relevant multimedia content.[9]

In creating rich topic pages a "global" vocabulary across the subsites becomes practically essential. BBC used DBpedia to unify the various vocabularies they were using on the different subsites, meaning they created *mappings* between the tags and the DBpedia-URIs, which was done automatically through DBpedia label lookup and context-based disambiguation (see chapter 2). This enabled them to interlink the content across the sites, because they were able to fetch content via a single DBpedia-URI. As a result they had the ability to generate rich topic pages, which also included data from external sources, discussed in the section on *Semantic enrichment*.

Fortunately VG already uses the same tags across the subsites, and the creation of richer topic pages is purely a matter of prioritizing it. However, interlinking subsites this way represents a highly valuable way of using Linked Open Data that it can nonetheless be useful to outline here for demonstration purposes.

3.3.1 Challenges

Since using a Linked Open Data vocabulary as a "global" vocabulary requires mapping, the challenges with this approach are the same as in the first suggestion; a lack of Norwegian entities in the knowledge base to map to. Thus some tags might not get mapped, or alternatively the missing entities can be added into the knowledge base.

Another potential challenge could be that the current tagging is too sparse, meaning that the mapping will not provide as many opportunities as it could. A solution to this could be employing a tool like DBpedia Spotlight or another knowledge extraction tool that returns DBpedia-URIs, that could tag the articles again. However, tagging multimedia content like videos could be difficult as the textual descriptions are usually very short, in which case improving the tagging would require further measures.

3.4 Enable third party utilization

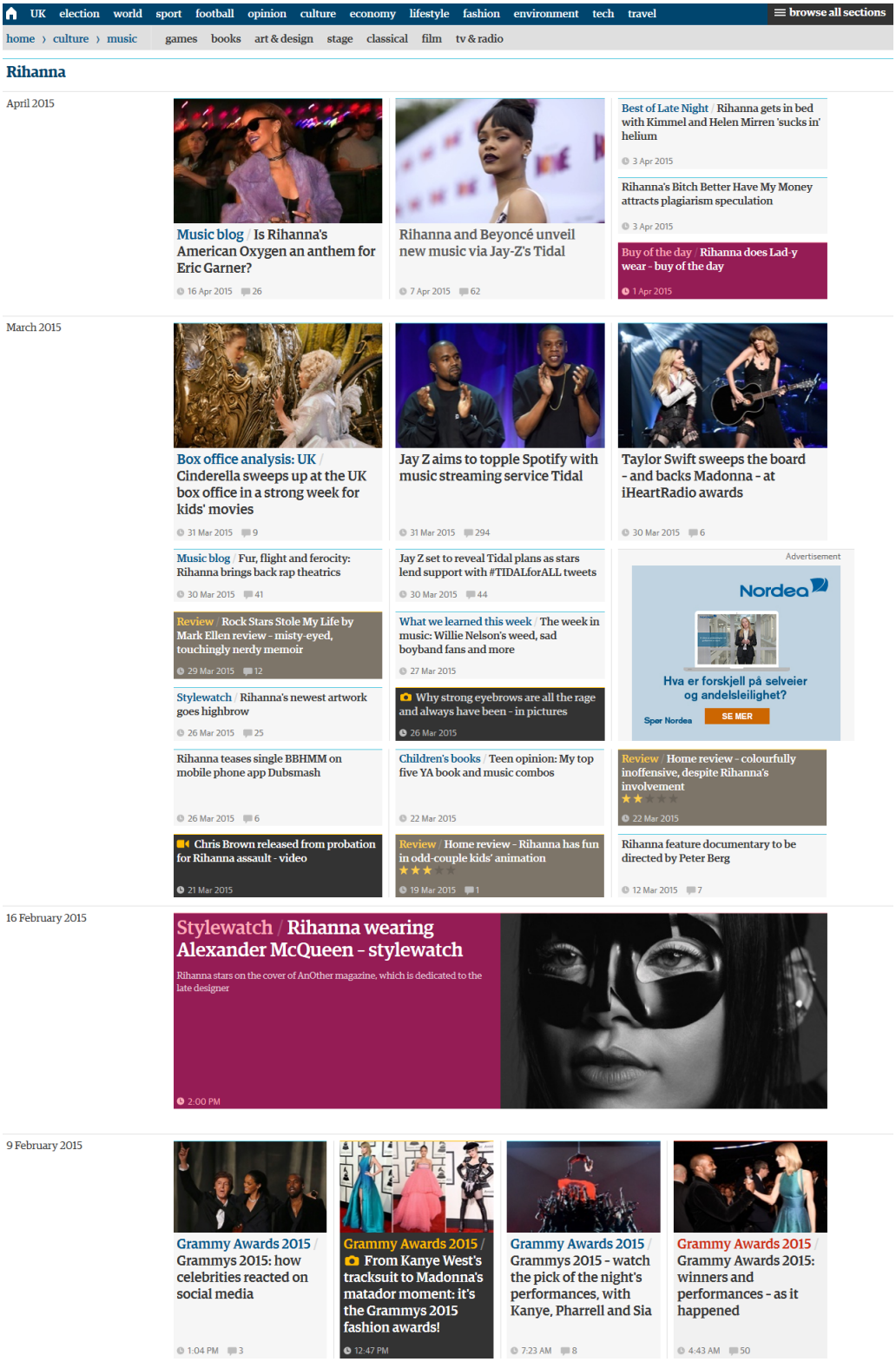
The standard way of accessing content from a website like VG.no is through browsing their web pages. To support navigation between the various pieces of content, the site includes certain navigational elements like a main menu, related links, and sometimes a topic page. These are all ways of improving the user experience to the reader. In some cases, however, a third-party might want to access the content — not by browsing through the HTML-documents, but programmatically. There could be many reasons for this, e.g. wanting to display links to a certain type of articles on their own page, or produce data from the different types of

[9] http://topics.nytimes.com/top/reference/timestopics/people/o/barack_-obama/index.html, viewed 8 April 2015

[10] http://www.theguardian.com/music/rihanna, viewed 8 April 2015

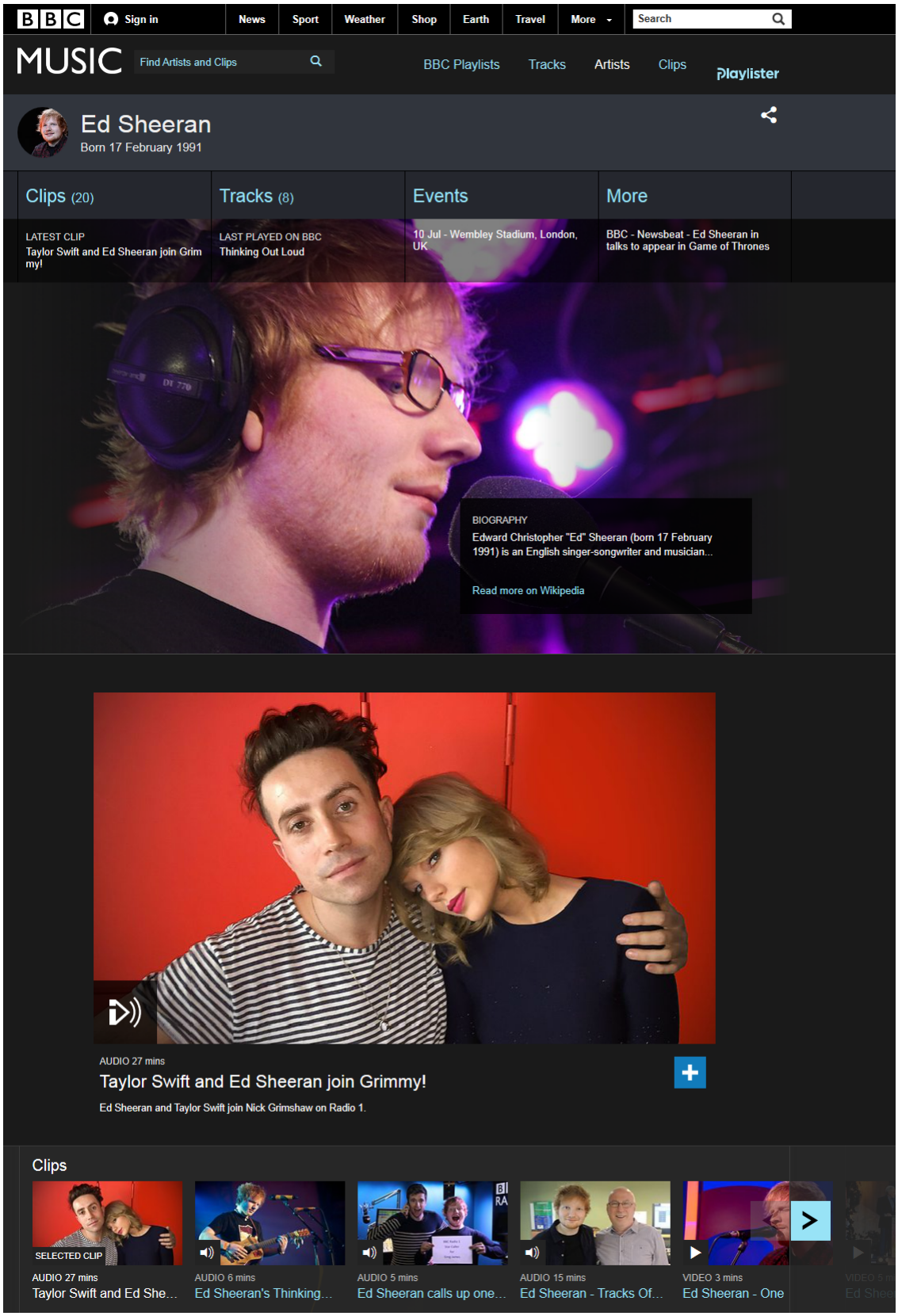
[11] http://www.bbc.co.uk/music/artists/b8a7c51f-362c-4dcb-a259-bc6e0095f0a6, viewed 16 April 2015

Figure 3.4: The Guardian's topic page on Rihanna



The Guardian's topic page on Rihanna include multiple different types of articles, picture galleries and other mutlimedia content.[10]

Figure 3.5: The BBC's topic page on Ed Sheeran



The BBC Music pages have successfully linked their contents to other BBC subsites. This topic page on Ed Sheeran include video clips, tracks, and events.[11]

content. A task like this entails creating a script or program that will get content by asking the server for a response that's not necessarily in HTML, like a browser usually does, but other formats that are easier to handle for a developer. And instead of having to use the navigational elements set up by the web designer, it's possible to set other restrictions or filters on the contents returned. For a web developer, this offers more flexibility when dealing with large amounts of content, especially when they only want certain parts of the content, e.g. the headline or a link. One way of easily accessing content for a third party developer is through making a request to the server through an *API*. An API, or Application Program Interface, is a collection of functions that allows you to access functionality or data within another system or service. In most cases, there is already an existing API that the page itself uses, and a company will simply *open* the API to outside actors. Opening the API entails removing some of the restrictions that were previously put on clients asking to access the server, and many choose to publish instructions online to facilitate correct usage.

Retrieving the correct content from an API requires the developer to have a certain amount of knowledge about the underlying database. Websites that handle large amounts of content normally use a database for storage, which in a news publisher's case would be storing articles and other multimedia content. In order to effectively retrieve the right content, each item has a unique ID in the database, often called the *primary key*. This id, however, is only meaningful in that particular database. E.g. an article with the ID 3465 is not necessarily the same as an article with the same ID in a different database belonging to someone else. The same goes for tags and categories, which will usually have unique ids in addition to their name. One of the main goals in Linked Open Data, however, is providing universal ids, which are in the form of URIs. If VG mapped its content, like the tags and categories, to Linked Open Data identifiers, a developer could use the universal ids to aid content retrieval. This would mean storing a mapping between the local ids and the Linked Open Data ids, and developers being able to use either one to retrieve content. E.g. if a third party wants all the articles about Beijing, which are tagged with tag 486, they could avoid having to find the tag id, and instead use the DBpedia URI and ask the server for content belonging to http://dbpedia.org/resource/Beijing.

The Guardian and NY Times are two large news publishers who have open APIs available[12] [13]. Both of these companies have some of their data mapped to Linked Open Data identifiers, e.g. The Guardian's album reviews are available through MusicBrainz IDs, and their book reviews through ISBNs.

The great advantage of enabling third-party utilization is the possibility of external actors driving traffic to the site, without having to pay for advertising. The payoff might not be as immediate as with the other suggestions, but the long-term effects could be a great benefit to the

[12] http://open-platform.theguardian.com/
[13] http://developer.nytimes.com/docs

organization.

3.4.1 Challenges

Opening an API isn't necessarily difficult in itself, but there are other factors to consider when outside parties suddenly gain access, like security issues. As with any online publisher, there are also certain legal issues that needs adressing. VG does not permit external actors to reuse their article texts for commercial use, so the content available through the API would most likely be the same as through an RSS feed.

The challenges regarding the mapping are the same as mentioned previously.

3.5 Semantic enrichment

Semantic enrichment means displaying information to provide contextual or general additional information. Providing context was one of the main points outlined in the leaked NY Times mentioned in chapter 1. Having readers going outside the site to seek additional information might be losing them traffic, and as a result this is one of the areas that news publishers might aim to improve.

There are many ways this is done today in news publishing. A common one is including the contextual information as part of the article text (*brødtekst*). During my observations in VG, I witnessed that many journalists were copying paragraphs from older VG articles on the same subject in order to reuse content. The paragraphs were often more general information about the event or person, e.g. what a person was known for. In the breaking news department, this was typical for follow-up articles on an event, and for the Entertainment journalists it was usually about adding more information about celebrities.

Another common way of providing context is through infoboxes. Infoboxes are one of the context-providing features that have been transferred from printed to digital format whilst remaining almost exactly the same in the process. In VG these are created manually by the journalist and saved in the publishing system, DrPublish, which in many cases have multiple infoboxes available on each entity. The publishing system keeps track of how many times each has been used, and many of them have only been used once, alltough sometimes eleven or more are available on the exact same subject. In other words, there might be potential for improvement on how these are created and managed. Linked Open Data could aid the creation of these infoboxes through providing data on the entity in question. As long as the entity exists in the knowledge base, data like date of birth, place of birth, occupation etc could be directly imported and displayed to the journalist while constructing the infobox.

Topic pages can also be used to provide context, like the ones mentioned in the section on interlinking. That section focused on merging content

produced by VG, but semantic enrichment would involve importing data from outside sources as well.

One of the benefits of using Linked Open Data to provide context is being able to do it automatically. However, there will usually always be some type of manual work no matter what, e.g. to fix issues like parsing errors. The Guardian chose to notify the users that the particular page they were visiting was created automatically by writing it in a notification box, and urging anyone who noticed an error to report it.

3.5.1 Challenges

One of the biggest limitations to using Linked Open Data for semantic enrichment is that there seems to be very little data on lesser-known Norwegian entities. This is discussed further in the Findings chapter, but in general information on many of the people and organizations relevant to the Norwegian media is sparse.

Another challenge it that the textual data, like abstracts or other short summaries of entities, is usually in English, and not available in Norwegian. Fortunately there is a lot of data that does not need any translation, e.g. numeric values like birthdates, coordinates (which could be used to import maps), or images, links or multimedia content.

Furthermore, any data imported needs to be highly reliable, as VG is dependent on distributing correct information. It would also have to be editable, both the data in the knowledge base and once the info is imported. Due to this, a local version of the knowledge base might be the best solution for keeping control of the data.

3.6 Reasoning to produce data on content

Sometimes the goal isn't to extract the content itself, but be able to say something about the content's characteristics on a higher level. Whether it's through mapping existing tags to Linked Open Data identifiers, or using the them as tags themselves, the structure of Linked Open Data means that it's possible to reason on the content. *Reasoning* in this case means, for instance, the ability to state that one entity has a specific kind of relationship to another entity, and consequently also a relationship to another entity. E.g. California (http://dbpedia.org/page/California) is located in the United States (http://dbpedia.org/resource/United_States), which in DBpedia is expressed as:

```
dbpedia:California dbpedia-owl:country dbpedia:United_States .
```

United States is a member of NATO, stated in DBpedia through the category http://dbpedia.org/page/Category:Member_states_of_NATO relationship:

```
dbpedia:United_States dcterms:subject category:Member_states_of_NATO .
```

The Netherlands (http://dbpedia.org/page/Netherlands) is another country who is a member of NATO, which is also expressed in DBpedia. This means that if VG has three articles about California (meaning they are tagged with "California"), and two about the Netherlands, it's possible to reason that VG has at least five articles about places that are members of NATO.

This type of functionality is most likely not useful to regular users, but is instead targeted towards data analysts or data journalists, and possibly third parties, e.g. media researchers. Being able to place content items into multiple specific or more general categories could mean e.g. the ability to identify the themes of the top 100 best selling articles in VG+, which is similar to what the German news agency in section 2.4.4 did in identifying trending topics. In that case they also developed functionality for viewing news within a particular region using GeoNames as a knowledge base, which is another example of how to use the data collected.

One could be able to state that 40 percent of the articles written about Asia in 2014 were about North Korea, or that there are 30 percent more articles about a certain political party this election than the last. In other words, it could be used to generate data on the content's characteristics as of today, but also what it was like in the past, and how the characteristics have changed. Information like that could be useful simply for the information's sake, such as in enlightening the organization about the nature of the content, or aid decision making, e.g. what future content should be about to maximise traffic on one of the specialized subsites.

3.6.1 Challenges

Reasoning can extract very complex data, but is dependent the quality of the existing tagging. If the tagging is insufficient, a knowledge extraction tool could be used to find a higher number of tags, and/or more suitable tags. It's also dependent on a sufficient ontology to reason from, so the correct information is being generated. It's reasonable to believe that a well-known knowledge base like DBpedia or Wikidata will be adequate.

3.7 Contextual information for journalists

For obvious reasons, the reliability of the content published is of vital importance to any news publisher. Doing research is an important, though sometimes time-consuming, task to journalists, and for good reason. While the story itself is sometimes from another news agency like NTB, more basic information like age, number of inhabitants etc. can come from websites like Wikipedia. Given that many of the infoboxes in Wikipedia are available in DBpedia, it is possible to get these simpler facts imported into an application. This could either be an external application, or directly into the publishing system as a plug-in.

One of the great advantages of making a semantic enrichment application for journalists as opposed to readers, is that the textual information

can be in English, which most of the Linked Open Data is. This opens up a lot more doors for the functionality of the application, but at the same time it might be less valuable in terms of keeping visitors on the site.

This type of application or plug-in could recognize what entities are mentioned in the article text, which could be done using a knowledge extraction tool. Next, it could search for the entities in a knowledge base like DBpedia, GeoNames or Wikidata. It could then extract certain basic information, like age and place of birth.

3.7.1 Challenges

As mentioned previously, knowledge bases seem to lack a lot of Norwegian entities, and thus the application would not be able to extract knowledge on these.

Another challenge is identifying what kind of information journalists want and need, which requires cooperation from this part of the company as well. Furthermore, there are many different kinds of journalists, who write different kinds of stories for different subsites. Each journalist might need different types of information, and not everything is available as Linked Open Data.

An application like this could either use the web service/API of a knowledge base, or VG could download the data dumps locally. The disadvantage of using web services is being vulnerable to server issues like downtime etc. However, downloading data dumps locally could take a lot of space, depending on the size of the knowledge base. In the case of different types of the data needed being available in different knowledge bases, the plug-in would have to send queries to multiple places. If it uses web services, this could potentially make it slow. If the data dumps are available locally, it would mean downloading two different knowledge bases.

3.8 Fact-checking tool

Taking the previous idea further, a "fact-checker" could be developed. As a plug-in in the publishing system, the fact-checking functionality could check simple facts in an article, either while it's being written or once it's finished. The plug-in would have to use text processing tools to identify statements that should be checked, e.g. a person's age, then find the date of birth as Linked Open Data, and identify whether the age in the text matches the age found as Linked Open Data.

Statements to be checked could be identified using Named Entity Recognition via a knowledge extraction tool, or defining certain heuristics, e.g. that any numbers between two parenthesis that occur after two words with capital first letters is an age-value, and that the two words in front make up the name of a Person-entity. It could subsequently search a knowledge base for a URI with a name-relation matching the name in the article, and return the birthdate-property value. It would then calculate

the age of the person, and then the difference between the two age-values, deciding whether it's correct or not.

3.8.1 Challenges

This is a fairly complicated and technically challenging suggestion. Making effective heuristics is no easy task, and they would have to be tested thoroughly. With this kind of functionality there's always a risk of false positives and false negatives. While a lot of false negatives means that the functionality is used less, too many false positives can make it annoying to the user. If the functionality is only applied optionally, the users have to be motivated. For motivation to exist, it probably has to work well.

Another challenge is deciding what facts should be checked. This depends on whether there are specific kinds of factual information that are mentioned consistently in articles, which can be identified by a heuristic, and furthermore are available as Linked Open Data.

This chapter has outlined many of the ways VG can utilize Linked Open Data. As part of this Master's thesis I've decided to develop a tool for providing contextual information for journalists, which is the suggestion from section 3.7. The prototype should provide journalists with additional information about the entities they are writing about. The aim is to simplify researching, and display data that could be used in the article or for further research. The functionality will be available as a plug-in in their publishing system named DrPublish. The following chapters describe how I did exploratory research to figure out what the journalists needed, and how this information was used to develop an early prototype. I then describe the process of evaluating the simple prototype, followed by the development of the actual plug-in. Finally are the results from the second usability test, which is the evaluation of the finished product.

But first, I outline the methods used in conducting the research.

Chapter 4

Methods and methodology

This chapter outlines the various methods I've used in my research. There were three rounds of data collection in total. For the first round, which was the exploratory research, I used interviews and observations to collect data, in addition to grounded theory to aid analysis and data collection adjustments along the way. After analyzing the data, I developed a prototype which was evaluated using formative usability testing. With the results from the usability test, I developed the prototype as an actual plug-in in the publication system. The results from the usability test that followed, in addition to all the other findings, can be found in chapter 6.

4.1 Methods for exploratory research

4.1.1 Interviews

One of the fastest ways of finding out something from a person, is simply asking. Interviewing is a commonly used technique in many research fields, not to mention in Human-Computer Interaction. Lazar et al. (2010) underlines how interviews are suitable both for exploratory research, requirements gathering, and evaluating prototypes when working with users.

In empirical research, however, interviewing is not necessarily as straight-forward as it sounds. Firstly, one has to decide how much structure the interview should have. *Fully structured* interviews have little flexibility, and means that the researcher have all the questions prepared beforehand and sticks to them in their prescribed order. The interviewer should not stray from the prepared questions, or comment or follow up the interviewee's comments. Another option is the *semi-structured* interview, in which the interviewer has a prepared set of questions, but has the flexibility to ask follow-up questions and explore to a greater extent. In an *unstructured* interview, the interviewer has only topics or a few questions prepared, but is free to follow them as he or she wants. In this type of interview, the interviewee has more control of where the questions go, as opposed to the two other types.

Secondly, the researcher have to decide how to do the interview. One

of the advantages of interviews is that they can be conducted in multiple different ways, like face-to-face or via phone. With the emergence of the digital age, it's also possible to do via e-mail, online chats or other instant messaging tools, or even video chats.

As opposed to administering questionnaires, interviews allows the interviewer to ask follow-up questions and go deeper if he pleases. Although it's harder to reach the same amount of people as with questionnaires or survey research, the ability to go deeper can provide richer data, and thus eliminate the need for more participants.

However, if the research requires the researcher to gather data from many people, interviews can be a lot of work to conduct and might not be the best option. Similarly, analysing interviews can take a lot of time, especially as they become more unstructured.

4.1.2 Observations

Another way of collecting data about people is through *observation*. Observations can be used in both qualitative and quantitative research, and can be executed in multiple different ways (Cozby 2008).

One of the common types of observation is *naturalistic observation*, sometimes called *field observations* or *field work*, which is when the researcher conducts the observation in the milieu of the participants. The researcher is thus *in the field*. This is great for describing and understanding the setting of the people you're researching, because the researcher gets to experience the setting himself to a certain extent. In a naturalistic observation, the researcher can go as far as become a participant himself, called *participant observation*. In this kind of observation, the researcher will take an active role in the setting and become part of the "group". In this way he has an opportunity to yield rich data through experiencing the setting of the participants firsthand. The danger of immersing in the situation to such an extent is becoming overly subjective, meaning not being able to think grasp the bigger picture, or take other people's view, due to your own experiences in the field. This is less of a concern in *nonparticipant observation*, in which the researcher maintains his role as an external actor. This allows him to observe the natural context, but more easily keep the data collection objective enough.

There are certain downsides to naturalistic observation that are hard to counteract. One is that it's often time-consuming, both the data collection and the analysis. The researcher does not necessarily know which data will be important or not, which can result in collecting very large amounts of data. Furthermore, the data collection has to fit the time schedule of the participants or phenomena, which isn't always the most convenient time for the researcher.

Another form of observation is *systematic observation*, in which the researcher collects data about a quantifiable phenomena, and is only interested in a few specific parts, e.g. one or more behaviors. Often the researcher will have developed an hypothesis prior to the observation, which the research is based on. In a systematic observation the researcher

needs a *coding system*. E.g. if he or she is studying behaviors, it could be as simple as "Active" and "Resting". However, it needs to describe the phenomena with enough detail for the hypothesis to be confirmed or falsified. Sometimes it can be a good idea to use existing code systems, as you are have existing research on how well these have worked previously.

One of the disadvantages of systematic observation is that it doesn't necessarily allow the researcher to go as "deep" as he wants. Coding will also sometimes require equipment, e.g. if you choose to video record participants to be able to do the coding later. In the case of having multiple people code the data, there should also be taken measures to assure that each coder use the codes the same way to increase the reliability of the results.

4.1.3 Triangulation

Triangulation means conducting research using multiple methods, like interviews *and* observations (Maxwell 2005). The belief is that combining two or more research methods will counteract some of the disadvantages to each one, meaning that the data you might not collect reliably using one method, you will get using another method. This way the researcher aim to get a more correct picture of what is happening. However, triangulation does not automatically increase validity. This depends on which research methods are chosen, as some research methods have the same disadvantages, like the self-reporting biases that might occur in interviews and questionnaires. Observation and interviews can be a good combination of methods as each is able to reveal aspects the other might miss — while observation is great for *describing* behavior, interviews give the researcher an opportunity to inquire about the underlying reasons and motivations.

4.1.4 Grounded theory

Grounded theory was developed by Barney G. Glaser and Anselm L. Strauss in the 1960s, and was inspired by the tradition in Chicago Sociology at the University of Chicago. Chicago Sociology used fieldwork and in-depth interviews extensively, while also emphasising social changes and their directions.

Essentially, Grounded theory is a method that permeates both the data collection and the analysis (Charmaz 2005). Traditionally many researchers have separated the collection from the analysis, choosing to do one first, and the other second. There are many pitfalls to this approach, one being the risk of the analysis becoming overwhelming, especially for new researchers delving into ethnographic studies with large amounts of qualitative data (Maxwell 2005). Grounded theory aims to avoid this by making the data collection and analysis simultaneous, which also means being able to let the analysis guide the collection along the way. For the researcher, this entails constantly being ready to abandon previous ideas

about how the data collection should be performed, and instead be guided by the data.

Just as many researcher choose to divide the data collection from the analysis, many methods involve predefining an hypothesis about the outcome. This hypothesis guides the data collection, and is usually related to a theory. In Grounded Theory, the researcher should not have a predefined hypothesis or ideas about what the analysis will reveal. The idea is to let a theory emerge from the data through the iterative process of analysis and data collection.

The analysis is done through *coding*. Coding means prescribing "codes", which in theory could be virtually anything. E.g. "Subject A was met with an error and sighed" could be coded with "Frustration". The researcher will usually choose a *coding paradigm* (Strauss 1987). Examples of coding paradigms are *strategies*, *consequences* etc.., which result in various categories of codes. The point of the open coding is to gain a certain understanding of what is really going on, which are shaped by the codes and the categories emerging from them.

The next step is *axial coding*, which involves investigating each category deeply and discovering relationships between them. This is usually not done in the early stages, but becomes more prominent as the researcher draws links between the codes and categories during the analysis.

4.2 Usability testing

Usability testing is a way of evaluating a system or a prototype (Lazar, Feng and Hochheiser 2010). It can be done in many stages of the prototype development, using anything from a paper prototype to a fully functioning system, and is usually conducted with representative users. Additionally it can be done on various types of devices, like traditional PCs, smartphones or tablets. In other words, usability testing can take many different forms, but the goal is always to improve the prototype.

Although usability testing is not always considered "research", they often utilize many of the same methods, like observations, interviews and questionnaires. Yet while traditional research design often aim to generalize, usability testing aim to identify and fix flaws in the prototype. This could be understood as while research design is used to *understand* a phenomena, usability testing aims to *evaluate* a solution.

There are multiple types of usability testing. Three of these are outlined below.

4.2.1 Formative usability testing

The goal of a *formative usability test* is to identify any design- or functional flaws that the user is able to discover through using a simple paper prototype. This is a type of usability testing done at an early stage to rule out any obvious mistakes that can be avoided before developing a higher-fidelity prototype (Lazar, Feng and Hochheiser 2010). It's usually

an informal process and focused on how the interface is perceived. Doing usability testing with very low-fidelity prototypes is intended to make the participant more comfortable with giving feedback and constructive criticism, since they can see that having to change elements or components won't be a very costly or long process. It's also beneficial for designers to get feedback at an early stage, especially in a user-centered design process, to avoid investing too much time and effort in design decisions that will ultimately be changed.

4.2.2 Summative usability testing

Summative usability testing is a more formal kind of usability test than the formative type (Lazar, Feng and Hochheiser 2010). During a summative usability test, the goal is to collect metrics and gather results that are statistically significant (Dumas and Fox 2007). The results are then compared to the company's pre-established needs. Overall, there's a much greater focus on quantitative measurement, and conducting the usability in a controlled environment. Participants are typically given a task list that will measure e.g. error rate, task completion time and user satisfaction. Qualitative data is usually also collected in some form, for the participant to express his or her thoughts less restrictively.

4.2.3 Thinking aloud

Thinking aloud is a method with what seems like a self-explanatory name, but is alas more complicated than it sounds. In its essence, "thinking aloud" means having the participant speaking/saying his or her thoughts out loud while they're interacting with a computer system (Jørgensen 1990). The goal is to identify errors in the system, and through gaining "access" to the mind of the user, attempt to understand how he or she encounters the errors and why. This information could be valuable to the researcher in designing the system in a way that users will understand it more easily and intuitively, and hopefully decrease the amount of errors a user makes.

The method originated from cognitive psychology and was often used in *introspection*, which is when subjects report their cognitive processes (Boren and Ramey 2000). It's thus called a *verbalization* technique. Later it became a valued method in usability testing, though the primary rationale of uncovering system errors is quite different from its application in psychology.

Many of the usability studies using Thinking aloud use the guidelines outlined by Ericsson and Simon in *Protocol analysis: Verbal reports as data* from 1984. Ericsson and Simon divide verbalizations into three levels; level 1 are verbalizations that need to "transformation" to speech, meaning that he or she simply reads something. Level 2 are verbalizations that need some kind of transformation, e.g. more abstract concepts and images. Level 3 verbalizations appear when the participant engage in more demanding cognitive processes, like filtering information, recalling events from long-term memory etc., or when he or she or is influenced by outside forces in

a way that might alter the thought processes. Ericsson and Simon only value the first two levels as actual data, as the third level verbalizations have been altered to a greater degree by social constructs or other factors that does not pertain to the actual system. They also recommend to avoid recording feelings, daydreams etc..

In Ericsson and Simon's opinion, the researcher should not interact with the participant beyond what's strictly necessary. The necessary interactions include instructing the participant in detail on what to do during the usability testing, which should be to ask him or her to simply verbalize their thoughts, but not necessarily *explain* the process. The researcher should also prompt the participant to remember to do so if the participant is silent for more than 15–60 seconds. Otherwise, the researcher should not intervene.

It has since been reported that researchers seem to apply the technique in very differing ways. For instance, the prompts to keep participants "thinking aloud" can range from short to long, be personal or impersonal etc. Furthermore, the timing of the prompts vary, especially since many researchers do not set a specified prompting interval. In general, the level of interaction between the participant and researcher seem to deviate the most from Simon and Ericsson's guidelines. This could due to software errors, the user getting stuck etc. Another reason for the discrepancies seem to be the varying contexts of the usability studies, which the researcher feel tempted or obliged to adapt to. Although understandable from a social standpoint, it makes results from some studies difficult to compare.

This chapter has provided an overview of the various methods that are used in the research for the prototypes. While interviews, observation and grounded theory belong to the "traditional" researching domain, usability testing is focused on evaluating a system or a prototype. Observations, interviews and grounded theory were used in order to discover the researching needs of the journalists in VG. After developing a prototype based on the findings, I evaluated it using formative usability testing. Next, the results from the formative usability testing were used to develop the plug-in. The plug-in was then evaluated using summative usability testing and Thinking Aloud.

The next chapter explains how the prototypes were built, both the low-fidelity paper prototype, and the actual plug-in. The design and functionality of each of these were based on the results from the findings presented in chapter 6.

Chapter 5

Prototype

With the help of the findings from the exploratory research, I was able to design a low-fidelity prototype on paper. This is presented in the following section, along with some paragraphs about my thought process around the design. After usability testing it, I developed a plug-in in VG's publishing system called DrPublish. The development process and technologies used are presented in the section called "High-fidelity prototype".

5.1 Low-fidelity prototype

After the exploratory research, I used Balsamiq[1] to produce a visual mockup to serve as a low-fidelity prototype.

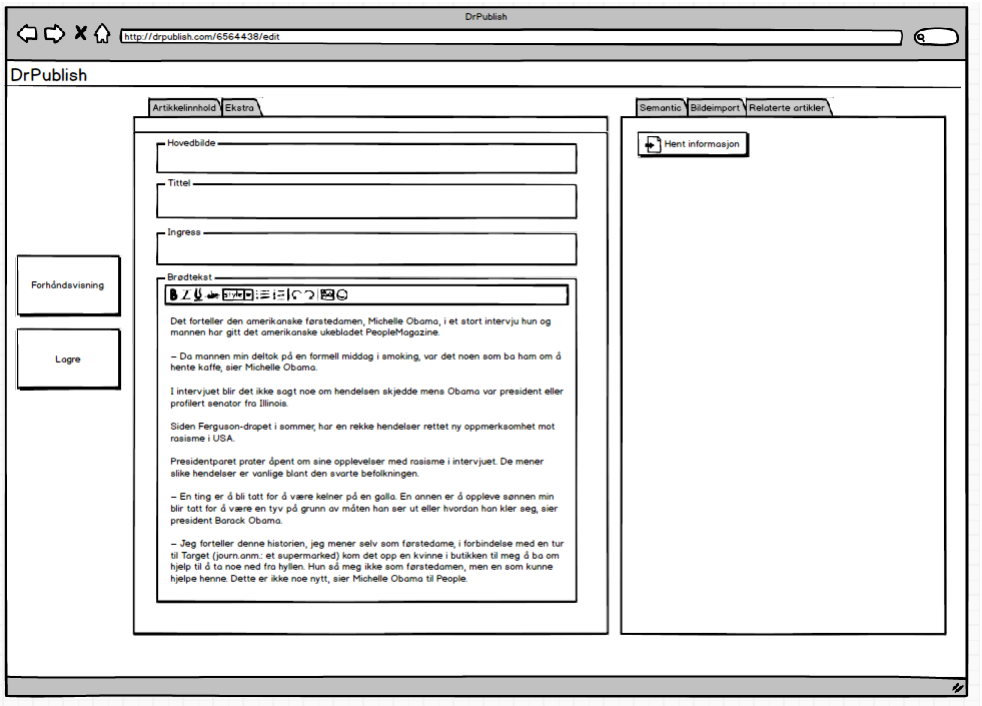
In the design-phase I realized that some factors were important to take into consideration, like the graphical user interface of DrPublish. Fortunately DrPublish is especially suited for integrating "locally" developed plug-ins, and VG is currently using several on a daily basis, like "Bildeimport" and "Relaterte artikler". I added them into my prototype to illustrate this, see 5.1.

Another consideration is what kind of data is available as Linked Open Data, compared to what kind the journalists need. As the Findings chapter will show, there was a certain overlap between these two. I decided to narrow my scope and focus on providing information about people and places. For people, the information it shows is the full name, date of birth, age, social media accounts, a link to wikipedia and its NY Times topic page, see figure 5.2. I also chose to include a picture of the person, for the journalist to make sure that the information showing is in fact about the person they're referring to. The image does not belong to VG, and can not be used for any other purposes. For places, the plug-in shows a map, containing a link to Google Maps, the link to its Wikipedia page, its NY Times topic page, Google Maps, and other local media sites. All of this information is available through either DBpedia or Freebase/Wikidata.

I purposely avoided some of the information that the journalists needed, but was the least reliable. Typical examples include relationship

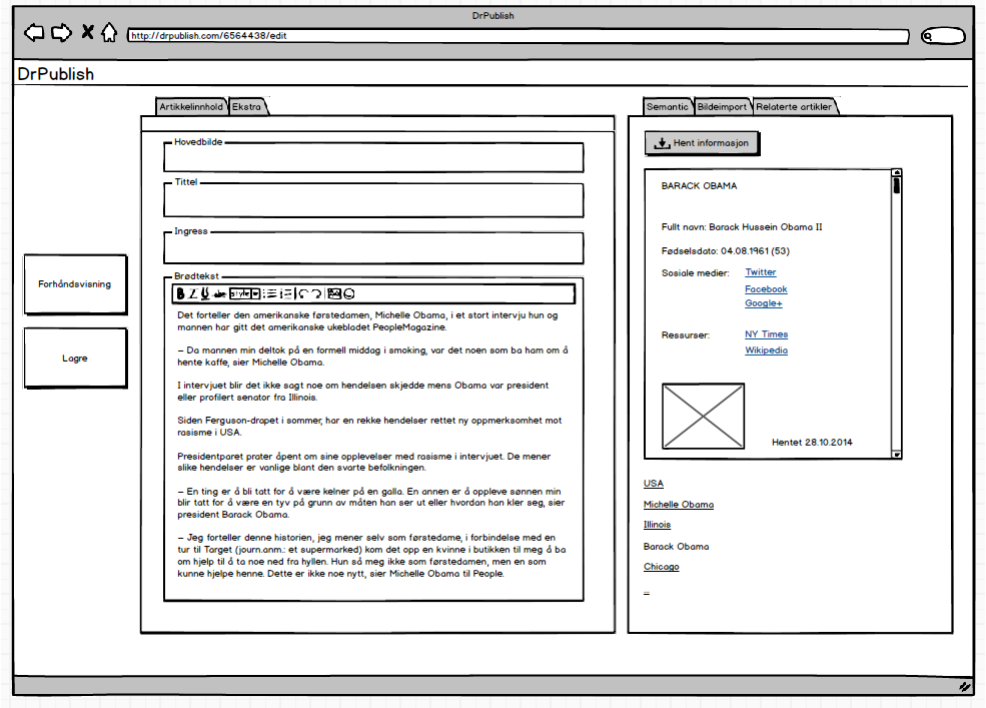
[1] http://www.balsamiq.com

Figure 5.1: The start screen of the low-fidelity prototype



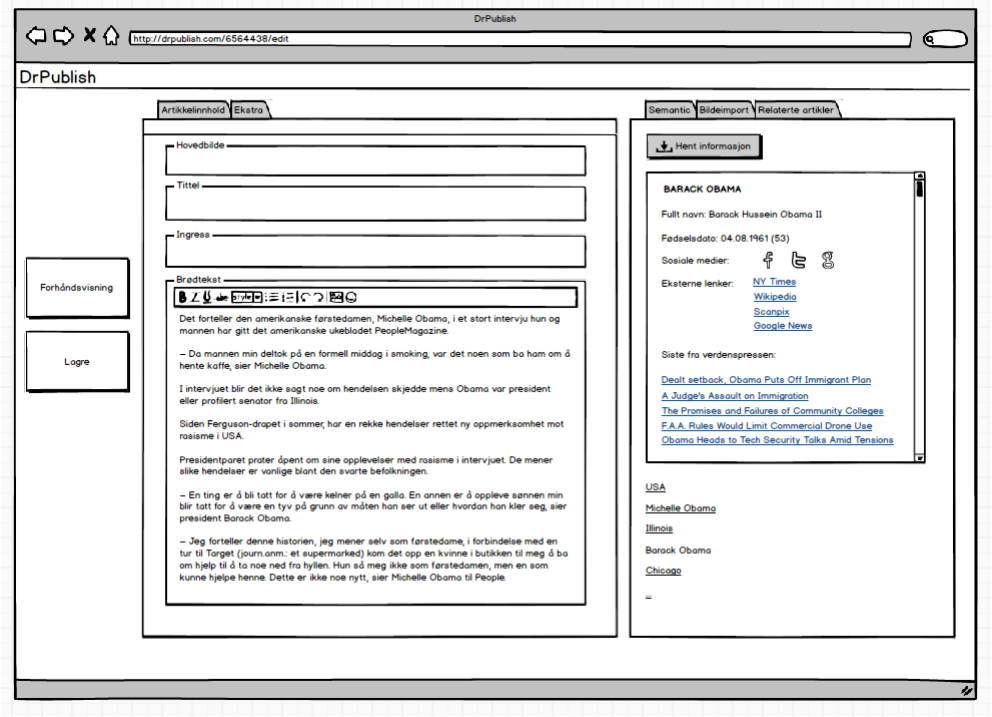
The start screen of the plug-in when it has been loaded into the user interface displays a "Get information"-button. Upon clicking the button, the user receives the various entities that have been found in the text. When clicking on one of the search results, the plug-in should display information on the entity.

Figure 5.2: The low-fidelity prototype displaying information about an entity



The plug-in displays information about Barack Obama, including his full name, date of birth, age, social media accounts, and links to his Wikipedia page and NY Times topic page.

Figure 5.3: Revised version of the low-fidelity prototype



The new screen of information about Barack Obama. After the formative usability testing I revised the design, and got the idea to instead of linking to the NY Times topic page, import the links for the most recent articles on the person, which is available on the topic page.

status and financial information, as these are variables that tend to change often. Unfortunately semantic knowledge bases like DBpedia does not always contain the latest Wikipedia-entry, nor does Wikipedia always have the correct information.

Reliability of the information is a hugely important consideration, as VG is dependent on providing reliable information to its readers. Since my research showed that the journalists were not using Wikipedia for anything else than very basic facts, the scope was naturally narrowed by this.

As my research was done primarily with News and Entertainment journalists, the prototype naturally aims towards these types of articles. Articles on international news will probably get the most information, due to lack of Linked Open Data specific enough for journalistic purposes in national news.

5.2 High-fidelity prototype

After the initial round of usability testing, I made an improved version of the prototype, including elements the journalists wanted but that can be technically challenging (see figure 5.3). These worked as guidelines for the developing of the DrPublish plug-in.

I had no previous experience in making a plug-in in DrPublish, but I

was offered the source code for a another plug-in that dealt with some of the same elements as mine would, which I could modify and extend. This was very helpful since it contained the paths to the various APIs etc., and imported some useful libraries like jQuery[2]. jQuery is a JavaScript library used for a multitude of things, like aiding the manipulation of different elements on a web page.

The plug-in was made using HTML[3], CSS[4], JavaScript and PHP[5], though I had very limited experience with these technologies. Fortunately the plug-in is simply an iframe within DrPublish, meaning that it's essentially a web page integrated in DrPublish, and doesn't necessarily have to interact with the DrPublish page elements. This made the coding part easier since it doesn't require you to learn another API, but on the other hand requires more styling to make it look and feel like a part of DrPublish.

Figure 5.4, 5.6 and 5.5 are screen shots of the final result.

5.2.1 The plan

The plan was to develop a first version of the plug-in that would provide information on persons — meaning it would recognize Person-entities in the text and display information useful to the journalist, and which was retrieved as Linked Open Data. The information I chose in the low-fidelity prototype was available through DBpedia and Freebase, two large knowledge bases. Both of these have endpoints set up, and a web page for executing your query and receiving the result in your web browser, which was particularly handy for testing various queries.

5.2.2 The tools and knowledge bases used

DBpedia Spotlight Web Service

DBpedia Spotlight is open source and available for download, but also offers a web service that is easily available, as mentioned in chapter 3. As this was intended to serve as a simple prototype, using the web service seemed like the best option. Switching to the actual software will however make the plug-in faster, since it wouldn't have to send the requests to an external address.

DBpedia endpoint

DBpedia, mentioned in previous sections, is one of the largest knowledge bases available, and contained many of the data I was looking for. DBpedia take queries in SPARQL, either in their HTML endpoint [6] or URL-encoded as a POST-request. I had worked with the endpoint on previous projects and knew it has had trouble with some downtime. I did not have issues

[2]https://jquery.com/
[3]http://www.w3.org/html/
[4]http://www.w3.org/Style/CSS/Overview.en.html
[5]http://php.net/
[6]http://dbpedia.org/sparql

with it this time, but it could also be downloaded in data dumps. That would also speed the plug-in up.

Freebase endpoint

Freebase is another large and well-known knowledge base in the Semantic Web community. However, Freebase is being merged with Wikidata in June this year (2015), and thus the endpoint will eventually become unavailable [7]. The data was not yet availale in Wikidata, so I did not have another choice. If the plug-in is to keep sending requests to Freebase, the data dumps have to be downloaded. The queries to Freebase are slightly different as they require MQL instead of SPARQL. MQL is short for *Metaweb Query Language* [8] and has a very different syntax to SPARQL, see below.

5.2.3 How it works

The code consists of an index-file in HTML, two JavaScript files in which app.js is the controller, and an API-folder consisting of four PHP-files that put together and execute the HTTP-requests. The plug-in initially displays a button, and upon clicking it, the plug-in will process the title and main story using the Article API, which is used to enable communication between the plug-in in the iframe, and the interface in DrPublish. The text is then sent to the DBpedia Spotlight endpoint using cURL[9] in PHP. The POST-request to DBpedia Spotlight was configured to only identify Person-entities, and returned the result (the entities) as a JSON-object. JSON[10] is short for for JavaScript Object Notation, and is a format made to be easy to read for both humans and machines. The object contained all the various Person-entities and their URI in DBpedia. The JSON-object is parsed in the main JavaScript-file, and the search results are displayed as little tags. Upon clicking one of the tags, the plug-in will first send a POST-request to the DBpedia endpoint. This is done by making a query in SPARQL using the entity name as an argument, URL-encoding the query, setting the preferred format to JSON, and executing the request. The query sent to DBpedia when asking for information on an entity, in this case the URI for Barack Obama, looks like this:

```
PREFIX dbp: <http://dbpedia.org/resource/>
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dbpprop: <http://dbpedia.org/property/>

SELECT ?name ?dateOfBirth ?wikipage
```

[7] https://plus.google.com/109936836907132434202/posts/3aYFVNf92A1, viewed 4 May 2015
[8] https://developers.google.com/freebase/v1/mql-overview
[9] http://curl.haxx.se/
[10] http://www.json.org/

```
WHERE {
    dbp:Barack_Obama dbpedia-owl:birthDate ?dateOfBirth .
    dbp:Barack_Obama foaf:isPrimaryTopicOf ?wikipage .
    dbp:Barack_Obama dbpprop:name ?name .
}
```

The results from DBpedia is the person's date of birth and English wikipedia page. The result of the request is sent to the JavaScript file, is parsed and displayed to the user. The person's age is calculated from today's date, and is displayed alongside the birthdate. Following this, a new request is made to Freebase asking for a person's social media accounts and NYTimes Topic Page. The queries for the NY Times topic page, continuing using Barack Obama as an example, looks like this:

```
{
"id":"/en/barack_obama",
"key": [{
        "namespace":"/source/nytimes",
        "value": null
     }]
}
```

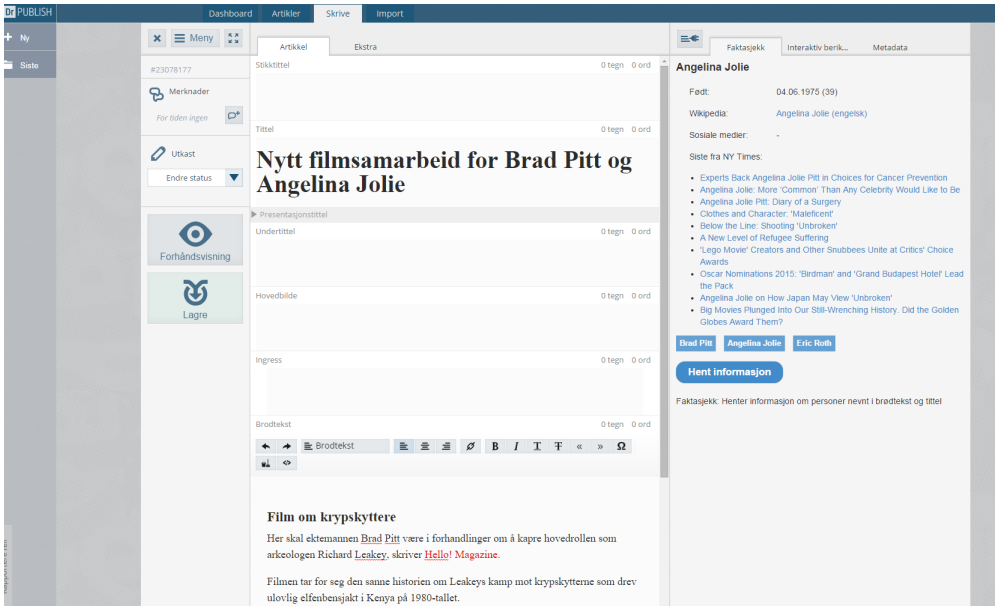
During testing I discovered that Google will start sending an error after a certain amount of API-calls, and ask you to sign up on their Google Play Developers Console[11] and register your project. Even after registering your application as a project, Google imposes certain usage limits[12]. The results from Freebase are parsed in the JavaScript-file, and the Facebook, Twitter and/or Instagram accounts are displayed to the user. These seemed to be the most prevalent social media accounts that journalists were using during the observations. The link to the NY Topic Page was used to execute a new HTTP-request, which was a GET-request to the given topic page that would get the HTML-file of the page, also called *screen scraping*. Each topic page has links to the latest articles the NY Times have written on the topic, and the goal was to display these to the user. Since the articles on the topic page were all given the same html-class, they were not hard to extract. The elements extracted were thus the headline of each article, and the links. These are presented as a list in the DrPublish plug-in.

This chapter outlined two prototypes: the low-fidelity paper prototype developed in Balsamiq, followed by the high-fidelity prototype which used DBpedia and Freebase as knowledge bases, and DBpedia Spotlight to recognize entities. Both of these were usability tested, and the findings are presented in the next chapter.

[11] https://code.google.com/apis/console

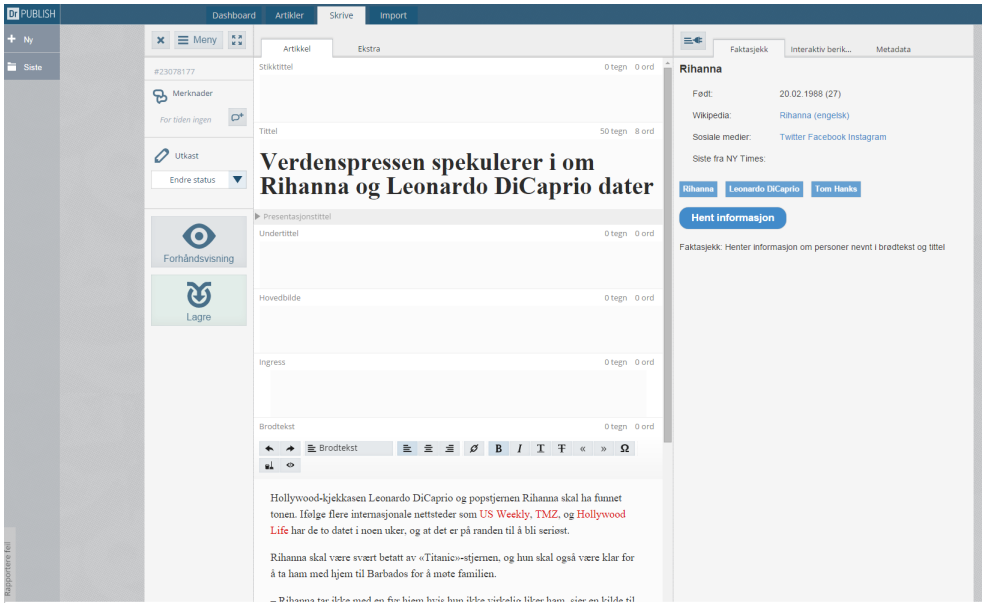
[12] https://developers.google.com/freebase/usage-limits, viewed 2 May 2015

Figure 5.4: The plug-in: Information on Angelina Jolie



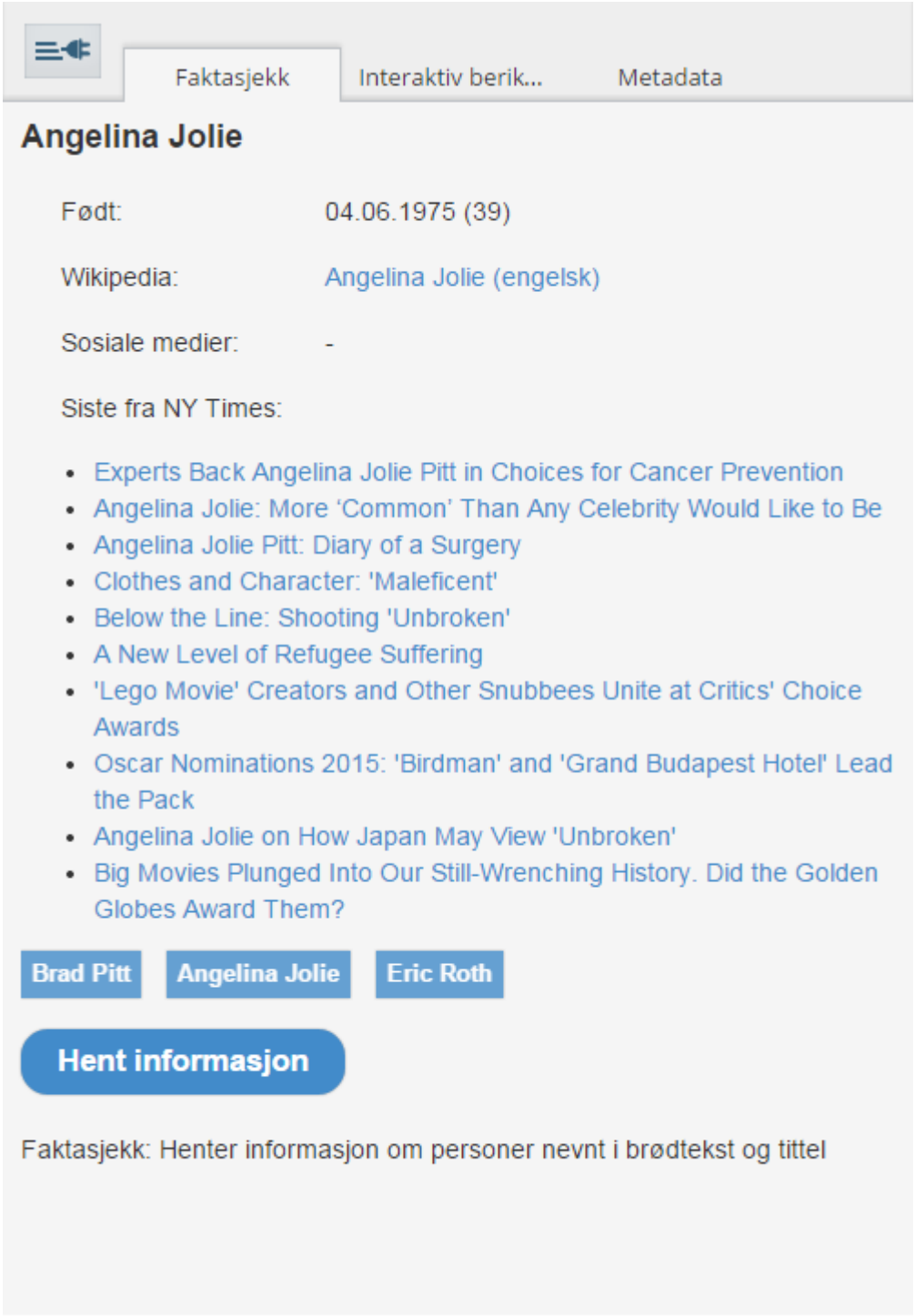
The publishing interface of DrPublish that the journalists use when writing an article. The article contents are written in the middle part, while the right part, or sidebar, has the plug-in imported. In this case the user has pressed the "Get information"-button, and clicked on "Angelina Jolie" when the search results were ready.

Figure 5.5: The plug-in: Information on Rihanna



When the plug-in doesn't find a link to the NY Times topic page in Freebase, it's not able to extract the latest articles. In the case of searching for "Rihanna", no NY Times link was available. It does, however, find links to various social media accounts.

Figure 5.6: The plug-in: Close-up of the infobox



The plug-in displays the date of birth and a link to the corresponding Wikipedia page, which are both extracted from DBpedia. It didn't find any links to social media accounts in Freebase, but was able to extract the link to her NY Times topic page, and displays the links to the latest articles.

Chapter 6

Findings

Using the methods outlined in chapter 4, I did three phases of research: the exploratory research, the formative usability test, and the final usability test with the high-fidelity prototype. This chapter explains how I applied the methods, and what the findings were.

6.1 Exploratory research

Prior to developing the plug-in, I did some exploratory research to investigate the journalists' needs for the functionality I wanted to develop. In cooperation with two of my advisers in VG from the development department, we decided that I would develop a plug-in that would supply the journalist with supplemental information as the article is being written. In order to decide which information it should display, I had to do some exploratory research.

My data collection was done through conducting interviews and observations, and the research questions were as follows:

- Are the journalists doing research outside the primary source for an article?
- If yes, from what sources?
- If yes, what kind of information are they looking for?

6.1.1 Conducting the exploratory research

The interviews

The interviews I conducted were semi-structured, as it allowed me to ask follow-up questions and explore their responses to a greater degree.

I did not find it necessary to audio record the interviews, since they were meant to be short and only have questions that didn't require long answers. They were conducted in a separate room away from their colleagues, to make sure they weren't influenced by the possibility of others listening. It was also to avoid influencing other possible participants.

First they were asked about what department they worked in, and what type of content they produced for VG. I then proceeded to ask about their own data collection methods: what the research process is like, what online resources they trust, and what kinds of information they are looking for from non-primary sources. By "non-primary" I mean any source of information that is not the source of the initial story. The full interview guide can be found in Appendix A.

The observations

I chose to do naturalistic observations, so-called *field work*, as I didn't have much knowledge of journalistic research, and wanted to explore the phenomena more freely than a systematic observation would allow me. These were conducted at each journalist's desk while they were working as normal, with me sitting in a chair behind them and taking notes. I would take notes of most of what they did on the computer, with emphasis on the types of information they were looking for, along with how they got the information and from where. Each observation lasted for approximately 15 to 30 minutes.

The sample

There were no predetermined amount of informants, as the plan was to continue until there was very little deviation in the findings. This started to occur after the fourth observation, and thus I only did two more. As both the stress level and the amount of work to do varied greatly among the journalists, and would change quickly during the course of the shift, I was appointed the informants who were available at that particular time by the editorial manager in the newsroom. Setting up meetings with journalists during their shift was quickly deemed as almost impossible.

Analysis

After the first round of interviews and observations I adjusted the questions somewhat, as I got a better understanding of what journalistic work in VG entailed. After the fourth interview I began open coding (explained in the section on Grounded Theory in chapter 4), and used different codes for *sources*and *types of information*, e.g. personal data, dates etc.. I noted all the various terms mentioned that would fit into one or more of each of the categories. After performing the two last interviews and observations and using the open codes on the notes, I categorized the codes into larger categories. I skipped mentions of Wikipedia after the question in the interviews about using it as a source, since it was hard to tell whether they would have mentioned it had it not been used as an example. Following this, I did axial coding by connecting the different sources to the various types of information, in attempt to discover patterns. The findings are presented in the Findings chapter.

I chose Grounded Theory as the data analysis method as my data was qualitative, and foresaw that letting the findings guide my data collection would be a helpful way of gathering and analysing data. However, Grounded Theory has traditionally been deemed most suitable for very rich data, preferably collected through audio and video recording. Although my data has been much less rich, Grounded Theory has still been an effective way of analysing the data and discovering patterns.

6.1.2 Findings

The context

The VG offices are located at the well-known *VG-huset* in central Oslo. The journalists are seated in an open landscape, the desks placed in various group formations. Many of the walls have big screens, displaying various news channels. Each work station have two computer screens, a mouse and a keyboard, and the journalist will connect his or her laptop to these via a docking station.

I interviewed and observed mainly two types of journalists: News journalists working with breaking news (*realtidssirkelen*), and Entertainment (*Rampelys*) journalists. These two departments have very different ways of working according to my observations. The News journalists experience a much higher stress level, as their primary focus is delivering quality news as fast as possible. They are given events to cover by the editorial manager in the newsroom (*Nyhetsleder*), which is usually done face-to-face, via their instant messaging system Lync, or e-mail. In some cases they discover stories on their own and ask the editorial manager whether to pursue it. A major way of keeping up-to-date is through constantly monitoring and browsing the NTB news feed, looking for new stories or updates on previous cases. NTB (Norsk Telegrambyrå) is a syndicated news agency that VG subscribes to, which continuously posts news to their web site. DrPublish, the VG publishing interface or Web Content Management System, offers functionality for directly importing NTB news into an article — an approach which is frequently used. Following this, the designated journalist will edit the text, add suitable tags, go through other relevant settings (enable comments etc), choose a category, and finally publish it. Publishing the article, however, is rarely the last step. As more updates from NTB are posted, or the journalist finds more information from other sources, the case is continually republished with more updates and general contextual information. After each republishing, the journalist will look through NTB, other national and international news sites or social media. E.g. many use Twitter through tools like Tweetdeck, which is a feed with tweets from the Twitter accounts of your choosing, organized into you own customized categories [1]. The Twitter feeds of the journalists I observed usually contained tweets from district police departments and various national and international news sites. They also use tools developed in-house, like

[1] https://tweetdeck.twitter.com/

Hunter nyhetsrobot which provides an overview of the newest stories published on various news sites and their placement on each site, an indicator of importance and relevance. *Nyhetskarusellen* will load the front page of different news sites into the same web browser window each ten seconds or so, switching between them, and in this way resemble a carousel. In general, the environment of news journalists at VG is fast-paced, and they are required to be present multiple "places" at once; e-mail, NTB, phone, Lync, other news sites, in addition to being available to communicate face-to-face with colleagues and supervisors at any time during their shift. Thus multitasking seem to be a key ability among news journalists.

There are mainly two "states" the journalists seem to go back and forth between; namely *producing* and *consuming*. This is evident in the way they use the two computer screens on each workstation. One is consistently used for producing content, as in writing articles, and shows DrPublish or a Word document. The other screen has either NTB or another news site. Thus this seems to be as much a mental separation as a physical one, and seem to be very useful in managing the nature of their work.

The working environment of the Entertainment journalists seems less fast-paced, but involve many of the same challenges and strategies as in the News department. The journalists appear to use NTB much less, and instead rely on their own researching skills in order to find stories to pursue. The relevance of the stories are much less reliant on being published quickly, and instead the journalists spend time editing the text. Since the writing of an article doesn't involve directly importing from NTB as much, and many of their main sources are in English, the journalists spend more time correctly translating quotes and difficult terminology. The constant republishing common among the News journalists was not as evident while I observed the Entertainment department, and when articles were republished it was due to minor spelling errors etc.. Since the primary source of a story is not always a news agency, their research seem to involve a lot more fact checking and consulting multiple different sources. Other than these aspects, the two departments appeared very similar.

The journalistic researching process

On the question of whether they write articles directly in the publication system, almost all of them replied that they would usually write the article in Word first, and later copy it into DrPublish. The only cases they would write directly in DrPublish was articles on breaking news. Some reported that this habit was due to issues with the previous Content Management System, which they reported as unstable and difficult to use for the actual writing. Thus they used to resort to other text editors until the actual publishing was due.

The tables in this section present the findings from the exploratory research in the News and Entertainment department, both from the interviews and observations. Table 6.1 lists the most popular sources of information that were mentioned and observed. As the table shows, many of the sources mentioned during the interview was not observed during

the observation. These are most likely sources that either are not used that often, or is used when the participant is working in a different department or on other types of stories, as many of the participants worked in multiple departments.

The reason for the high occurance of Wikipedia-mentions could very likely be due me explicitly mentioning Wikipedia during the research question. For those who responded that they sometimes used it (which every participant did), I counted as an occurrence. However, only two were also observed using it as a source during the observations.

I also coded the different types of information they were looking for, outside the primary source for the case. These were recorded both during the interviews and observations. As seen it table 6.2, factual information on people were definitely the most popular, after searches for images. Next was language-related information, like translations and spelling, followed by paragraphs, information surrounding events, and links to previous articles.

Using axial coding, I mapped relationships between the different kinds of information and the sources (see table 6.3). Factual information on persons were most strongly linked to Wikipedia and existing VG content, while social media, colleagues, the police, international news sites and existing VG content were used to collect general additional info ("Events"-category). Language-related questions were answered either by colleagues, Wikipedia, Ordnett, or pointed out by a reader.

Implications for the prototype

The exploratory research revealed many relationships important for the prototype. The information a journalist needs is diverse; contact information (e-mail, phone number and address), personal information (date of birth, relationship status etc), social media content, translations, synonyms, links to other VG articles to use within the text, previously written paragraphs for reuse, and general additional info around cases. Thus a prototype should focus on one or more of these. The biggest limitation as to which I should choose is what kind of data is available as Linked Open Data. Chapter 2 mentions some of the biggest knowledge bases available, and most of these contain information that either belongs to a very specific domain, like the movie industry, or aims to cover a very broad spectrum of things, like DBpedia. Other types of relevant Linked Open Data available is language-related, like the Norwegian Wordnet. FOAF, one of the large knowledge bases mentioned in chapter 2, contains plenty of contact information, but seemingly little relevant to VG. Social media content to a certain degree available as Linked Open Data in Freebase, but not consistently. The general additional information the journalists seek is usually too recent to be available as Linked Open Data, and thus simple factual information about people seem to be the best option.

The reliability of sources is of vital importance to news agencies. The sources they used during the observations were NTB, Ordnett, various social media accounts, various national and international news sites,

Table 6.1: Sources of information

Source	No. participants	Interview	Observation
American and other international news sites and blogs	5	4	4
Social media (Instagram, Twitter, Facebook)	5	2	3
VG archives or other existing VG content	5	5	2
National news sites	2	0	2
NTB	3	2	1
Scanpix	4	0	4
Wikipedia	6	6	2
Police or "politirunden"	2	1	1
Ordnett	1	0	1
Colleague	2	1	2
The Norwegian Yellow Pages	1	1	0
Press releases	1	1	0
Skattelister (public lists of earnings and taxes paid)	1	1	0
Tinglysninger (directory of public notices)	1	1	0
Store Norske Leksikon (an authoritative encyclopedia in Norwegian)	1	1	0
Surveys	1	1	0
Domain experts	1	1	0
IMDB (Internet Movie Database)	1	1	0
Getty Images	1	1	0
Kulturaktører	1	1	0
Main people involved in a case	2	2	0
A reader	1	0	1
External archives	1	1	0

A table showing the sources being used during the exploratory research. The "Interview" and "Observation" column represent the amount of participants that either mentioned the source during the interview, or I saw using the source during the observation. If a participant used a source, or a type of source, multiple times, it's only counted once in this table. The list includes both web resources and others.

Table 6.2: Types of information

Types of information	Examples	No. participants	Interview	Observation
Picture or image	Pictures for articles	4	0	4
Factual information about persons	Date of birth/age, name, relationship status, family relations, other background info	3	2	1
Language-related	Translations, synonyms, spelling, meaning of a word or a term	3	0	3
Paragraphs	For re-use from previous VG articles	2	0	2
Events	When, where, general additional info	2	0	2
Links to previous articles	Links to be used within the article text	2	0	2
Contact information	Phone number, address, e-mail address	1	1	0
Quote	To be used in an article text or title	1	0	1

The above table lists the types of information being searched for, organized into categories, along with the number of participants using them during the observation, or mentioning them during the interview. The counting was done in the same as with the sources of information; if a participant looked for the same type of information twice, or mentioned it twice, it was only counted once. The Events-category is the broadest one, and includes instances where the journalists would look for information about when or where something happened, i.e. information about an event or happening. It was often in relation to a story about a person, e.g. where a celebrity was during Christmas.

Table 6.3: Results from the axial coding

Information type	Source	Participant uses
Image	Scanpix	4
Events	Social media	2
Events	International media	2
Language-related	Colleague	2
Paragraphs	Existing VG content	2
Events	Colleague	2
Factual information about persons	Wikipedia	1
Factual information about persons	Existing VG content	1
Language-related	Wikipedia	1
Language-related	Ordnett	1
Language-related	Reader	1
Contact info	Politirunden	1
Events	Police	1
Events	VG	1
Events	Main people involved	1

Results from the axial coding, sorted by highest occurrence. For some of the mentioned occurrences of either a type of information or a source, I wasn't able to use for this type of coding. E.g. some mentioned that they would browse American or other international news sites, but not specifically what type of information they were looking for. Other times I would observe the participant use a source, without me knowing exactly why. Some of the names from the previous tables have been shortened here, e.g. "American and other international news sites and blogs" has been shortened to "International media". I also found it appropriate to separate "Politi" and "Politirunden" in this table, as these were used for different types of information.

existing VG articles, the Police, and sometimes Wikipedia for very basic info like date of birth or the spelling of names. The interviews revealed multiple other sources as well. The prototype should preferably use one of these, but not all are available as Linked Open Data. Only two seem to be viable alternatives, which is using Wikipedia through DBpedia, or Ordnett through Wordnet, although the last two are not the same. As DBpedia is one of the biggest knowledge bases available, and contains the factual information about people that the journalists seem to need, I considered it the best alternative.

Using the findings from the exploratory research, I developed the low-fidelity prototype introduced in chapter 5.

6.2 Usability testing the low-fidelity prototype

6.2.1 Conducting the formative usability testing

Since the aim of the first usability testing was to give the journalists an insight into what I had in mind so far, I made a paper prototype of wireframes developed using MyBalsamiq[2]. The full low-fidelity prototype can be viewed in Appendix B.

After signing the informed consent form, the usability testing consisted of give a brief explanation of my thesis, and then presenting the journalist with the prototype. The prototype was the Balsamiq prototype printed out on paper and stapled. Each page was turned by either the participant or me, and they were asked questions like: "What would you do from here?" and "Is this what you expected would happen after clicking that link?". I was intent on keeping the atmosphere light, and the usability testing more like a conversation. All of them were asked whether the information displayed would be useful, if anything was missing, and if the words were easy to understand. Lastly I asked for name suggestions, as I was unsure what would be an intuitive name that would describe the functionality.

6.2.2 Findings

I had five journalists test the low-fidelity prototype, and the reactions were overall positive.

When asked about whether the initial button ("Hent informasjon" / "Get information") was intuitive enough, some answered that they would like an explanation underneath that described the plug-in and what clicking the button would do. This was later added to the revised low-fidelity prototype.

When showing the users the NY Times-link, several pointed out that they would also like the last couple of articles written on the subject, from both VG and other news agencies. The NY Times Topic pages contain links to the some of the most recent articles on the given subject, but getting these would be more difficult as it would require "screen scraping" the topic page

[2]https://balsamiq.com/

and getting the links, instead of simply linking to the Topic page itself. It was, however, not as technically challenging as I first anticipated, and the final plug-in has this functionality implemented.

Another common suggestion was adding information about what the person is known for. This seems difficult as various predicates (i.e. links between a subject and an object in a triple, see chapter 2) serves this function for different people, e.g. in Wikidata Rihanna has an occupation-link (namely http://www.wikidata.org/wiki/Property:P106) to "singer", but Obama's occupation link points to "politician", though the link "position held" as "President of the United States of America" would be more descriptive [3].

As to be expected, a lot of the ideas suggested by the participants were not possible to implement, or falls outside the scope of the thesis. Among these suggestions were to display the link to VG's topic page on the subject in the plug-in, which is difficult without a mapping from their own vocabulary to a Linked Open Data vocabulary, and without a mapping it unfortunately falls outside the scope of the problem area in this thesis. Another suggestion was to integrate information that changes very quickly, which I wanted to avoid in fear of displaying incorrect information and thus give the impression that the other information is not to be trusted. Other suggestions were about including other types of information, which unfortunately aren't available as Linked Open Data, like links to articles from CNN or other news agencies.

Other various feedback included changing the term "Resources" to something else, and including links to searches in Scanpix and Google News.

In addition to all the suggestions, I also have to make sure to keep the plug-in visually simple, and not overflowing with information. Deciding what information is important, and what information is *even more* important, is a hard one to make without further usability testing.

One somewhat worrying, but very valuable, input was that the usage of the plug-in might be limited by the fact that the journalists usually use two screens; one for writing articles, and one for doing research. Including links to external websites in the plug-in means disrupting this separation between the *producing* and *consuming* of content, mentioned in a previous section.

After each usability test, I asked for name suggestions that would describe it accurately and suite its purpose. The suggestions were: *Faktaark*, *Bakgrunn*, *Assistenten*, *Nyttig informasjon*, *Infohenter* and *Informasjon*.

[3] I later discovered a suitable predicate for this information in DBpedia, namely dbpprop:shortDescription, see chapter 7

6.3 Developing the prototype

6.3.1 Lack of data reliability

After deciding on the functionality for the DrPublish plug-in and discovering what information would be useful for the journalists, the next step was finding out whether any of that data was available as Linked Open Data. This, however, was not the only restriction. From browsing through a lot of DBpedia and Freebase pages, it became evident that some of the information available as Linked Open Data is incorrect, usually because it's outdated. This often boils down to the nature of a specific variable, like marital status. For DBpedia, the reason seems to be that it hasn't updated that specific Wikipedia page recently enough, and it thus doesn't contain the latest information. Fortunately most DBpedia-pages include a variable stating which date the data on that particular page was extracted, which could give the user an indicator of the reliability of the data. I considered reliability to be very important as the plug-in would be used by journalists, who's job is to provide reliable information to the public, and will not use a tool displaying information they would have to double check anyway. Thus, I avoided extracting variables that are prone to changes and fluctuations. I also discovered a service that aims to avoid this, called DBpedia Live, by doing the translation from the Wikipedia-page to RDF-data (like the data in DBpedia) only once a request for the entity has been received. However, from experiences with downtime and other limitations with web services, I knew VG would most likely prefer to download the data dumps locally, and using the regular DBpedia endpoint thus provides a more realistic impression of how that solution would work in practice.

6.3.2 Lack of data in Norwegian

While exploring the alternatives for what data would be suitable to display in the plug-in, it became very clear that there's a general lack of Linked Open Data in Norwegian. For instance, I was not able to find a Norwegian DBpedia, although it's available in many other languages (Mahdisoltani, Biega and Suchanek 2015). This is somewhat less problematic considering that the only users who will view the information are journalists, who are more than likely familiar with the English language. It could, however, be a considerable limitation in implementing semantic enrichment, as mentioned in chapter 3.

6.3.3 Lack of data on types of entities important to journalists

Upon testing DBpedia Spotlight using their web demo and through my plug-in, many well-known Norwegian entities did not generate any hits. There could be many reasons for this; either that the entity didn't have a corresponding DBpedia URI, shortcomings in DBpedia Spotlight making it unable to find it, or the plug-in or web demo not working properly. In some cases, I wasn't able to find the entity in DBpedia, and in other

cases I would get a hit in the web demo but not in the plug-in. While it's plausible that in some of the cases, the underlying reason is the code I developed, but it's still a very real issue that the English Wikipedia, and thus DBpedia, lacks information on a lot of Norwegian entities. This is unfortunate as many of the people VG writes about are Norwegian, and not necessarily well-known enough internationally to foster an infobox in the English Wikipedia.

Another category missing from DBpedia are people who for have become famous very recently. In these cases, Wikipedia is usually quick to get updated, while DBpedia will be updated later. Even when NY Times generates a topic page within a short amount of time, it's only available through their own API since knowledge bases like DBpedia and Freebase/Wikidata are slower. Furthermore, using a knowledge extraction tool will not render a search hit on the person, because he or she doesn't necessarily have a Linked Open Data URI yet. This means that getting the identifier, and thus the lastest articles on the case from NY Times, is dependent on that Wikipedia generates a page on the person, then DBpedia crawling the page, and then hopefully someone adding the NY Times Topic Page link, unless the query for the latest articles are done through the NY Times API.

6.3.4 Using English knowledge extraction tools for Norwegian texts

Just as there was a lack of Linked Open Data in Norwegian, I was not able to find any knowledge extraction tools tailored to the Norwegian language. However, this became less problematic as I narrowed the scope to only include Person-entities. In developing the plug-in I was able to edit the settings in DBpedia Spotlight to only return URIs belonging to People-entities, which resulted in considerably less false positives. While testing DBpedia Spotlight in the web service demo, and in my plug-in, I discovered that the easiest false positives to get are the ones from common Norwegian words, like "er" (interpreted to refer to the TV-serier E.R.) and "fordel" (interpreted as Fordell Castle), which are the examples used in chapter 3. A potential solution to this issue is defining a list of "stop words", which are words that should be removed from the text before processing.

Note that I only tried using DBpedia Spotlight as a knowledge extraction tool, and none of the others mentioned in chapter 3. I didn't have a reason to believe that the other tools would do much better as they were also tailored for English texts. I also concluded that false positives might be better than false negatives in a plug-in like this, as the user himself chooses which Person-entities to import information on.

6.3.5 Multiple different query languages

Although the actual data in the various knowledge bases seem to follow the standards set by W3C (outlined in chapter 2), the way of extracting the data varies. DBpedia uses the query language SPARQL, which is one of the

most well-known query languages in the Semantic Web world. Freebase, as mentioned, uses MQL, or the Metaweb Query Language. Wikidata, another large knowledge base, have many different options for accessing their data, many of which are made by third parties. These seem to vary in terms of query language [4]. This means that querying a knowledge base through a web service or API can require you to learn a new language, and querying multiple can require learning even more. While some seem to resemble, others are very different.

A solution to this could be to download the data dumps from the knowledge bases and use a software like Fuseki [5] to set up a SPARQL endpoint locally. This means that instead of having to query the knowledge base in the query language defined by the given web service, you would only have to use SPARQL. Given a stable server, this approach could also help with the issue discussed next — downtime.

6.3.6 Downtime and limitations of web services

When using web services, there's always a risk of the server being temporary unavailable for various reasons, like maintenance. In earlier projects with Linked Open Data I had experienced some issues with downtime on DBpedia, but this has seemingly improved over the past year. When downtime does occur, which it usually will at some point, the plug-in will not display the date of birth, age and link to the Wikipedia-page. The user might get confused to why the plug-in suddenly doesn't work, which makes the plug-in appear unreliable. Fortunately DBpedia and most large knowledge bases let you download the data dumps and store them locally, which given a stable server can be a viable solution.

During development I also discovered that as a user of the Freebase API, you only get a certain amount of requests each day [6]. If you exceed the limit, you will receive an error urging you to register your project at the Google Developers Console page [7]. Google has put similar restrictions on their other APIs, including some of the Google Maps APIs in which you have to pay after a certain amount of requests per day[8].

Another issue is that Freebase is announced to be merged with Wikidata in June, meaning that after a while the plug-in will not be able to extract social media accounts and the NY Times links the way it does today. At the time of development, I was not able to find the data I wanted from Freebase in any other knowledge base, and unfortunately these data had not been transferred to Wikidata. Thus Freebase still seemed like the best option, especially since social media accounts were one of the information sources the journalists were using a lot.

Both the issue of request restrictions and the removing of Freebase could be solved by downloading the Freebase data dumps before they're

[4] http://www.wikidata.org/wiki/Wikidata:Data_access, viewed 3 May 2015
[5] http://jena.apache.org/documentation/serving_data/
[6] https://developers.google.com/freebase/usage-limits, viewed 2 May 2015
[7] https://console.developers.google.com/
[8] https://developers.google.com/maps/faq, viewed 2 May 2015

removed, or switching to querying Wikidata using another query language.

6.3.7 Getting help online

As most people who have done any kind of programming know, help through the forums like StackOverflow and blog posts are often essential in solving various programming problems. The downside of using a software like DBpedia Spotlight is that the help online was extremely sparse due to the fact that it's not very well-known. When getting an error that doesn't include a hint to any kind of solution, a Google search would retrieve very few hits. The hits were either from mailing lists or were many years old, and in many cases there were no answers to the question. Struggling with seemingly nonsensical errors can be extremely frustrating for developers, and not being able to receive help is an important disadvantage. Fortunately the software is open source, and the source code is easy to find on github[9]. Providing one has the time and skills, it's possible to delve into the DBpedia Spotlight source code and figure it out from there.

Getting help with DBpedia and Freebase was easier, though knowledge bases like these are still lesser-known technologies. Fortunately, my impression is that there seems to be a small amount of very engaged users doing a lot of helpful answering on forums like StackOverflow.

6.3.8 Crowdsourced data - lack of consistency

One of the main advantages of using Freebase in my plug-in was the large variety in what kind of data was available. Social media accounts, local newspapers and the NY Times topic page was among the variables I was interested in using, and which were missing from DBpedia. Unfortunately it was quite often that these data were missing on many entities. E.g. neither Justin Bieber or Rihanna had a NY Times topic page in Freebase, although both topic pages existed. Similarly many social media accounts were missing, and in many instances only contained one or two, e.g. a Facebook and a Twitter account, although a quick google search revealed that they had a highly active Instagram account as well. Thus it seems that there's a downside for knowledge bases trying to collect very heterogeneous data, because it can result in large inconsistencies in what is available for each entity. This means that although the user might be tempted to include a certain variable, he or she will have to take into account the fact that data on these types of entities might very well be missing in many instances.

[9] https://github.com/

6.4 Usability testing the high-fidelity prototype

6.4.1 Conducting the summative usability testing

The high-fidelity prototype was evaluated through usability tests. The usability study I conducted was inspired by summative usability testing outlined above.

My goal was to answer the research questions: *To what extent do journalists experience the experimental plug-in as useful?* and *How do the journalists experience researching using the plug-in compared to traditional researching?*

The summative test was conducted at each participant's workstation, as during the formative testing. I explained what the usability test would be like, and each signed an informed consent form. I proceeded to explain what the aim of the plug-in was, and introduced a scenario and a task list. The scenario was that the journalist is writing an article about Barack Obama meeting Rihanna, which the participants writes into DrPublish. The task list was as follows:

Use the prototype to:

- Find Barack Obama's age
- Find Rihanna's Wikipedia-page
- Find Rihanna's instagram account
- Find the last article the NY Times has written on Barack Obama

The tasks were not many, nor time-consuming, as the functionality is still very limited. The tasks were chosen on the basis of which functionality was available, and which I considered to be the most important, as is recommended (Lazar, Feng and Hochheiser 2010).

Although summative usability testing usually involve quantitative measurements, there were some limitations as to which would be suitable. Since the journalists were often interrupted, both when doing work and participating in my last usability test, any time measurements would be difficult to do, like the time used to complete a certain task. I also wanted to encourage the participants to "think aloud" while executing the tasks, and feel free to give feedback. Measuring time could potentially be disrupting the informal vibe. Instead I recorded the number of tasks completed and errors on each task.

After completing the task list, each participant was asked a series of questions regarding their thoughts on the functionality and design of the plug-in, e.g. *What do you think of the design of the plug-in*, *Is this useful information?* and *Can you see yourself using the plug-in in your work?*. I chose to do this as a semi-structured interview as opposed to presenting a questionnaire, since being one-on-one with the journalists had resulted in very rich data previously. Many of the journalists have seemed eager to share their thoughts and give feedback in the past, and information like that could easily be lost in a paper or online questionnaire. Furthermore,

conducting a semi-structured interview would be practical and feel natural after the observation at the workstation.

As with the observations, the usability testing was conducted at each participant's work station. Lazar et al. considers this to be ideal since it feels natural to the user, and he or she will experience the same attention limitations and other cognitive challenges as during the actual work. This is not the controlled environment described by Dumas et al., but on the other hand had the real-life elements that could still provide reliable results.

Both the quantitative and qualitative measures were recorded using paper and pen. Video taping might be able to collect data in a more reliable way, but runs the risk of feeling too intruding to the participants. Pen and paper kept the informal vibe that I felt was necessary to get the rich and honest feedback I wanted.

However, there turned out to be several issues with applying this kind of usability testing that I had not foreseen. Firstly was the influence of the setting, which was at each participant's workstation. Doing the usability testing in the open landscape meant entering a somewhat social setting, while I was attempting to collect quantitative data. The social setting made asking the journalist to do tasks feel unnatural, especially as most of the journalists would start talking and testing various names in the plug-in right after I had shown them how to import it, but before I had the chance to explain the task list. They were all very eager to give feedback, which was what they perceived to be my goal. Obviously any kind of research involving participants won't necessarily feel "natural", but me choosing to keep the research in the participants' everyday environment resulted in a bigger pressure to conform to the existing social setting. Additionally, completing tasks and counting errors felt wrong since the functionality in the plug-in was very limited. It also seemed counter-intuitive to count errors when a very large part of them were due to system bugs. Fortunately I was aware of many of them, although some were discovered there and then.

For these reasons, I ended the usability testing early and reevaluated my choice of method. With the help of my supervisor, I decided to apply Thinking aloud instead as it seemed more suitable for the setting and the plug-in.

6.4.2 Conducting Thinking Aloud

Simon and Ericsson outlines a somewhat rigid approach to applying Thinking aloud. From my previous experience with usability testing in VG, I knew that the context of the journalists' natural setting would have a big impact on both me and the participants, and thus I should ideally be able to to adjust to the social requirements of the situation, while still conforming to the guidelines. For that reason, I chose a less strict approach than Simon and Ericsson recommend.

Conducting the usability test started by having the editorial manager point me toward journalists that might have time to participate. After finding each participant and having the informed consent form signed, I

proceeded to explain why I was there and a little about the goals of the plug-in. I asked the participant to "think aloud", emphasizing that this does not necessarily mean *explaining* what they do or why, but rather what he or she is thinking. We then imported the plug-in into the user interface together, and I asked the participant to type in a well-known name, like "Barack Obama" to get them started. From there, the participants were instructed to verbalize their thoughts. Next, I encouraged them to try other names as well, like "Hilary Clinton" and "Vladimir Putin", to see if they disovered anything new or had any other comments.

My goal was to only intervene when it was necessary, or whenever someone encountered bugs. Although I knew from the observations that many journalists know how to "escape" bugs once they've encountered them, like refreshing the page, it felt necessary to explain that what they had encountered was a bug, and that they hadn't done anything "wrong".

After the session I conducted a short interview:

- How do you experience collecting information using the plug-in compared to your usual way?
- Is this plug-in something you could see yourself using (given it was bug-free)?
- The date of birth is extracted from Wikipedia. Do you find that problematic in any way?
- Is there anything particular about the plug-in that you like or dislike?

6.4.3 Findings

The final usability test was done with five participants in total, all of whom were working at the time, and seated at a workstation.

For some participants, thinking aloud came seemingly naturally. They verbalized what they saw, like "I see the age, the date of birth...". In other cases they would say to themselves: "Am I supposed to click this button?". Others remained mostly silent until prompted. For some, however, all the questions they voiced were most definitely directed towards me, and in those cases I would answer. As I had expected, the pressures of the social situation/context forced me to adapt the somewhat rigorous approach outlined by Simon and Ericsson, and thus some parts of the usability testing ended up resembling a conversation, with more of a journalist-developer vibe than participant-researcher.

In general, very few participants experienced any issues in using the plug-in. The most difficult part for many seemed to be actually importing it into the user interface from the list of about fifteen plug-ins, which wasn't actually a part of the usability test. This involves clicking a button next to the other plug-ins, and choosing *Infohenter* from the list. From my observations in the exploratory research I knew that all the journalists used at least one to three of these on a daily basis, e.g. for categorizing and tagging articles they wrote. However, the last used plug-ins each journalist

used are automatically loaded into the interface each time they log in, and some were not familiar with how to load other plug-ins. Another reason for this could of course be that my instructions were inadequate or misunderstood. However, explaining it became a lot more difficult when realizing that the journalist have no common term for what these are called. I decided to use the word "plug-in", but resorted to simultaneously pointing to the Import-button nevertheless.

Of the errors made during the usability test, many were due to bugs. One was that upon pressing a search result twice, the information (birth date, age, social media accounts etc) would appear twice on the page. This is an especially easy bug to stumble upon as the actual search is somewhat slow, and there is currently no loading icon or anything else indicating that the button has been clicked.

Another challenge was searching for Norwegian names. In the case of "Jens Stoltenberg", no NY Times article show up since the topic page hasn't been added to Freebase. Another participant tried "Tone Damli Aaberge". Upon getting no results, they tried "Tone Damli", in which they got a hit on "Francis Tone".

For internationally known people, the participants usually got the correct search results, e.g. for Barack Obama, Hilary Clinton, Vladimir Putin, Brad Pitt and Angelina Jolie. One tried "George Bush", which didn't render a result until adding the "W." in between.

Nevertheless, there were some unforeseen user behavior. One participant kept marking the name before clicking "Get information", which they did multiple times before each search before remembering that it was unnecessary. Upon attempting a new search, one participant tried loading the plug-in into the user interface again, instead of clicking "Get information" once more.

Another surprise was when one participant reported that she had actually already been using the plug-in (!), mostly to find people's age. However, she was not aware that in some instances the latest articles from NY Times would show up until I showed her.

Age information seemed to be the feature that the participants viewed as the most helpful. When I asked whether it was problematic that the data was extracted from Wikipedia, most of them said that Wikipedia were their primary source of age anyway. Others were more sceptical, and one suggested importing the age from two different sources, or an "official" source, just to be sure.

While many were also interested in the article links, every single participant had one question in common: "Why does it only show links from NY Times?". Attempting to explain that "they have published their tags as Linked Open Data" might not have actually answered anyone's question, but it quickly became clear that displaying links from *only* NY Times seemed almost counter-intuitive to some. Of the feedback, some said that in order to get the "full picture", they would use Google News as this provides them with news from many different sources (they can then cross-reference from different sources). Another participant pointed out that sometimes the latest news isn't what you need, but rather news of a

specific kind related to the person, or news that provide a more historic overview.

Out of the five participants, only two made suggestions for improvement regarding the user interface and functionality. One stated that the NY Times links to recent articles filled up too much space on the screen, and that both said they want to see more basic information, like what the person was known for or previous occupations.

Given that it would have less bugs, most of the participants stated that they would use the plug-in in their everyday work. The main concerns were the limitations for Norwegian entities, and their existing working habits.

Overall, the participants seemed genuinely interested in what I had developed, and were more than happy to give feedback.

This chapter presented the results from the various data collection methods I employed. These were used in developing the two prototypes from chapter 5. The next chapter discusses what these findings mean for the usability of the plug-in, and the use of Linked Open Data for this kind of functionality.

Chapter 7

Discussion

This chapter discusses the findings from the previous chapter, and is divided into the results from the development, and the results from the usability testing.

7.1 Developing the prototype

7.1.1 Data reliability

One of the main challenges I had during the design-phase of the prototype was finding suitable data. There were three requirements: (1) The data should be useful to the journalist on a fairly regular basis, (2) the data has to be available as Linked Open Data, and (3) the data should be reliable. By *reliable* I mean in terms of correctness and consistently being available.

I tried to uncover what data the journalists needed in my exploratory research, which were things like factual information about people (e.g. date of birth, marital status), language-related information (e.g. the spelling of a name or a word), contact information (phone numbers) etc.. I then attempted to find these data in knowledge bases, and next discover which of these data were reliable. I quickly discovered that although knowledge bases like Freebase had very heterogeneous data, like a person's romantic relationship, these data were often outdated. This simply comes down to how each individual knowledge base gather data. DBpedia extracts information from Wikipedia, but all the datasets are updated only with new releases of DBpedia[1]. The releases don't seem to come at a regular schedule, but approximately once or twice a year. Services that do the translation from the Wikipedia page to structured RDF-data live might be an option, but entails querying an external endpoint instead of downloading the data dumps locally. This means being vulnerable to downtime or slower functionality, and other disadvantages to using external services. I experienced the same issues in Freebase, in which parts of the data is crowdsourced. For this reason, I decided to focus on data possessing characteristics that would make it reliable by nature, like date of birth.

[1] http://blog.dbpedia.org/?p=77, viewed 4 May 2015

The way DBpedia and Freebase are updated also means that they don't always contain new entities that should be there. This was especially true for people who have become very famous in a short amount of time, which are people journalists typically want information on.

Another side-effect of the crowdsourced data in Freebase was that which types of data was available for each entity varied greatly. This made it hard to decide which variables that should be displayed in the plug-in, as I wanted the plug-in to be consistent in what types of data it displayed.

In general, this means that although knowledge bases contain very diverse data, and large amounts of it, not everything is relevant to news publishers, and the parts that are needed might not be reliable enough to be used, both in terms of data validity and consistent data retrieval.

7.1.2 Relying on external online services

The plug-in I developed sends requests to multiple different servers, both DBpedia Spotlight, DBpedia, Freebase and NY Times. While this was a quick and easy solution as a developer, it made the plug-in much slower than necessary. Many of the participants in the final usability test were unsure whether they had clicked the various button or not, because it would take a little while for the results to appear. Although it would have helped to add a loading-icon or another design element indicating that it was working, it would most likely be a lot faster if the knowledge base and the knowledge extraction tool was placed on a local server.

Placing the knowledge base and knowledge extraction tool on a local server also means being able to fix server issues in case it goes down etc., although this wasn't a big issue while I developed the prototype. Other benefits include being able to edit source code, avoiding server restrictions set by the host, and greater control of the data.

7.1.3 Using lesser-known technologies

Although this thesis demonstrates multiple different uses for Linked Open Data, the field itself is not very well-known. As a consequence, there seems to be less content available on the internet about the various technologies that are used in creating, maintaining and general handling of Linked Open Data. By *content* I mean tutorials, blog posts, forum posts etc. Sometimes developers have to rely on this kind of information, e.g. if documentation is lacking. Some errors are more abstract or difficult to solve than others, and in those cases forum posts on forums like StackOverflow can provide a lot of help. However, as the technology is not that well-known, some errors get very few search results. Furthermore, as the field itself is still small, there are less people that can answer questions on these forums. This means that delving into the field of Linked Open Data can be tricky for developers at first, since there is less help available. It also means that any developers actually learning Linked Open Data technology would do the community a great service in answering questions online, publish tutorials etc.

7.1.4 Different standards

In querying both DBpedia and Freebase in the plug-in, two different query languages had to be used. When doing further research on other knowledge bases, it became clear that there were multiple query languages being used in the community. This is an issue in using knowledge bases' web services, but is avoided when downloading the data dumps locally. Storing the knowledge base locally would most likely be the best solution for VG in any further use of Linked Open Data, and thus this challenge isn't necessarily relevant to VG. For small applications like plug-ins, however, it could be well worth the time to develop a normalization scheme between the query languages.

7.1.5 Being Norwegian

As I was researching Linked Open Data, one of my main concerns quickly became that the field is not tailored to many other languages than English. Understandable as it is, this is somewhat limiting for use outside English-speaking countries. This is especially true for knowledge extraction tools, because making these work properly with the Norwegian language requires tailoring, and not all knowledge extraction tools offer this kind of functionality. Many allows text input in other languages, but at the time writing this, none seem to support Norwegian. This wasn't huge a problem in the development in the plug-in, because I discovered that DBpedia Spotlight could filter the type of entity it returned. This removed many of the false positives compared to if the plug-in had required many different kinds of entities. This means that the tools aren't necessarily as flexible for Norwegian actors as English ones, and might require additional work, e.g. making workarounds.

A second disadvantage for VG is the lack of Linked Open Data available on Norwegian entities, and the lack of textual data in Norwegian. These challenges have been discussed many times in previous chapters, as this was a concern since the beginning. This placed certain limitations on the plug-in, which was unfortunate since the functionality was there, though the data was missing. These limitations are also highly relevant for VG in further use of Linked Open Data, depending somewhat on the specific case.

The plug-in and other tools using this technology will most likely be most useful for journalists working with international news or entertainment news, as these domains often involve internationally-known entities, whether it's people, events or organizations. Workarounds are harder to make as the problem is the actual data available, and thus the solution for VG is either to wait until the data is published, or publish it themselves.

7.2 Usability of the plug-in

7.2.1 Differing attitudes and skills

Ironically, one of the things that ended up revealing a lot about the usability of the prototype wasn't actually part of the usability test. When asking the participants to load the plug-in into the user interface, most seemed unsure about what I meant. There could be many reasons for this; no official terminology for what the plug-ins are called, poor instructions, or contextual factors. When continuing to the actual test however, it became evident that their understanding of importing plug-ins differed widely. While one participant wanted to import the plug-in once more when asked to do a new search, another participant stated that she had already been using it. Suffice it to say, this says a lot about the journalists' skills and attitudes toward technology. While one seem unfamiliar with the concept of plug-ins in DrPublish, another actively seeks out new ones available and use them, even without any instruction. Still, she had not been aware of the NY Times article links, meaning that giving out instructions when releasing new plug-ins might be a good idea nevertheless.

The difference in attitudes and skills among the journalists is valuable feedback regarding the usability of the plug-in, because it might reflect how they cope when faced with new technology. Not all the journalists know what plug-ins are, how they work or where to find them, which could mean that useful plug-ins might not receive the appreciation they deserve, my plug-in included. While all the journalists use various plug-ins everyday, the particular set of invaluable plug-ins they use are very ingrained in their working habits, and they already know very well how to use them. It's not unlikely that other plug-ins with widely different functionality become overlooked, also because the motivation for people who view new technology in any negative way has to be strong, which again means that the technology has to perform very well.

7.2.2 The power of habit

One of the advantages of having observed the journalists at work is that I was able to gain a certain level of insight into their researching habits. One of these were that they consistently use one screen for researching, and the other one for writing. In the researching screen they would open a new tab when looking for something new, and many used the Google plug-in to search. Using the plug-in I developed disrupts this clear separation between *producing* and *consuming* mentioned in chapter 6, but it's hard to tell whether it's a habit that was formed purely out of practicality, or if it reflects a mental separation between the two, and whether these are important or not.

Many participants in the usability testings mentioned habits as a reason why they possibly would *not* use the plug-in. For some it might have been a polite way of saying that they didn't like or didn't need the functionality. Others might have meant it, but nevertheless it was mentioned by multiple

participants in the usability tests.

7.2.3 Reactions to the functionality

All the participants reported that they liked the plug-in and the functionality it comes with. For some of the participants the best liked feature seemed by be age, which they said would save a couple of steps (like opening a new tab in the browser, typing the name etc). One said that age should always be behind a person's name, meaning that this feature was particularly valuable to them.

Other participants were more impressed by the links to the NY Times articles, which they also deemed as useful functionality. However, everyone reacted to the fact that there were only articles from NY Times, and not any other national or international newspaper. The drawback of only getting news from one source is that the journalist doesn't necessarily get the "full picture" of what the media is reporting, in spite of NY Times' good reputation. I wasn't able to tell how problematic this was, but it nevertheless means that combining news sources in the plug-in could be very valuable, and that researching ways of doing this is an idea for potential future development.

The biggest limitation of the plug-in seemed to be its lack of information on Norwegian entities. This is a serious drawback because many articles are about Norwegian people, and in many of those cases the plug-in will not render any results. This isn't just a problem in itself — the biggest drawback is that it makes the plug-in seem somewhat unreliable. It's possible that this will improve when DBpedia releases new versions, or if other datasets are released that contain Norwegian entities, like a DBpedia extracted from the Norwegian Wikipedia.

Another issue was the loading time, both for getting search results and getting the data for an entity. However, they didn't seem to mind it as much when they knew it was actually loading something, meaning that a loading indicator (symbol or text) should be added.

There are obviously advantages and disadvantages to using Linked Open Data. Some of these might be more difficult to overcome than others. Making a normalization scheme can be a relatively quick solution to deal with the different query languages, while more data on Norwegian entities might prove harder to attain. Furthermore, the community is still missing knowledge extraction tools equipped for handling Norwegian text input, although partial workarounds are possible.

Fortunately the feedback on the prototype was good, from both the journalists and the editorial manager. Whether the plug-in will prove useful to the journalists remains to be seen, as the usability test couldn't reveal the ultimate impact of technical skills and habits.

Chapter 8

Conclusion

This Master's thesis has explored the subject of using Linked Open Data within a major digital news publisher, namely VG. It has outlined the field and its characteristics, and suggested various ways the data can be used within the company. In pursuing one of the ideas, I had to conduct some interviews and observations which helped me in grasping the needs of the journalists. A prototype was developed and subsequently evaluated in two stages. This concluding chapter describes some of my experiences during this process, which had its ups and downs. Finally I present some thoughts on any further development of the plug-in, in addition to potential future work.

8.1 Writing for an actual company: My experience in VG

Writing for an actual company had its advantages and disadvantages, but most of all a lot of lessons.

My first challenge was in cooperating with the editorial department, consisting of the journalists and the editorial managers. I firstly had to explain my role as a Master's student, which is easy to understand in theory, but was harder to explain what entailed in practice. Furthermore, although they were all technically competent, explaining Linked Open Data to the editorial managers and journalists is no easy task. Naturally they wanted to know what I was looking for, but in explaining my research questions I seemed to create more questions than answers.

The difference in discourse was evident at other times aswell, especially during the actual research. Applying grounded theory allowed me to adjust the data collection along the way, which was much needed when I discovered that the world of the journalist was very different from what I initially thought, and that my preconceptions were reflected in my interview questions. At the time of writing the questions, I now realize I had a general lack of knowledge about journalistic writing and research. Although I conducted a pilot interview and observation first, adjustments had to be made multiple times. For instance, I decided to remove a question that was repeatedly misunderstood; "What kinds of cases do you write?".

It was always interpreted as what kind of themes, but it was meant as types of content, like articles, in depth-cases, reviews etc. Initially one of the questions asked them to describe the process of writing stories that only required online research, which I quickly discovered reflected my own outdated perception of their working life. In my head I'd had an image of journalists being out in the field and afterwards writing the case, just like in the popular media-depiction. This turned out to not be very representative for the journalists I interviewed, as the findings in chapter 6 reveal. I also ended up not asking about how much research a case took, as few journalists seemed to consider the word "research" to mean the same as I did, and thus it ultimately displayed my lack of knowledge of their working habits. Suffice it to say, entering the world of journalists in the two departments gave me a realization or two about the target group I was to develop for.

Exploring the world of the journalist profession was challenging not just due to the discourse, but also because of the natural urge to blend in to the world you've entered. While I was physically a part of the environment during the observations, I was still an outside actor, sitting behind them in a chair and taking notes. Needless to say, this was most likely as odd for them as it was for me. It probably didn't help the case that they seemed unsure of how I could possibly need the information I was gathering, which could be due to me not explaining it well enough. This, on top of the fact that they are extremely busy, left me with a nagging feeling of being "in the way". Necessary as it was, the exploratory research became an uncomfortable ordeal for me, teaching me that while able to produce very valuable data, field research can be almost like an exercise in bothering people.

Interestingly, the research seemed to become a lot more comfortable for both parties when I had developed an actual prototype, even when it consisted wholly of paper sketches. There are probably multiple reasons for this. Firstly, the social dynamic could have been different because both parties had a common object to focus on, instead of one watching the other. It also involved interacting, which probably felt a lot more natural than observing or being observed. Another factor can be that the common object, in this case the prototype, said something about my thoughts and where I was coming from, and they are likely much more familiar with the journalist-developer relationship than the participant-researcher kind. This probably made my role and what I was looking for much easier to understand for the journalists, and in turn respond to. While many seemed unsure of what I was looking for during the observations, giving feedback is most people are familiar with, and many even seemed *eager* to offer their expertise. This was also true for the editorial managers, especially when I produced the actual plug-in and demonstrated how it worked in DrPublish. My main contact in the editorial department even told me to let him know when he should send out an e-mail to all the journalists about the new plug-in, which stood a stark contrast to the feeling of being "in the way" I'd experienced earlier.

8.2 Recommendations for further development

It's clear from chapter 3 that there are many ways VG can use Linked Open Data in their company; for tag suggestions, as a controlled vocabulary, produce rich topic pages, semantic enrichment, enabling third party utilization, reason to produce data, contextual information or fact-checking. Each of these have their own challenges, though certain issues seem to apply to several. Some are due to the fact that the content is in Norwegian (lack of data on Norwegian entities, lack of data in Norwegian, no knowledge extraction tools tailored to the Norwegian language), while others are related to data reliability, lack of help online etc.

The prototype in the form of a plug-in demonstrates one way of using Linked Open Data in practice, and the findings from the subsequent usability testing yielded positive results.

If VG wishes to use the plug-in further, I have several recommendations.

Firstly, it retrieves data from Freebase, which is scheduled to be merged into Wikidata in 2015. This means that the data the plug-in retrieves from Freebase, which are the social media accounts and NY Times topic page, have to be extracted from Wikidata or other sources. Wikidata currently doesn't seem to have an official endpoint, but there are others available. Unfortunately none of them seem to use the same query language as Freebase, meaning that changing knowledge bases will also entail changing the queries themselves, not just the address. One option is to look into the NY Times API on how to get the topic page, or extract the recent articles directly from the API instead of screen scraping the topic page, if these are available through the API.

From the feedback from the usability test it became clear that the journalists would like to know what each person is known for. At the time of development I avoided this because I couldn't find a relationship-type that would accurately capture what the journalists were looking for (discussed in chapter 5). I've since discovered dbpprop:shortDescription in DBpedia which points to a short description of the entity. Another option is to use dbpedia-owl:abstract and only use the first few sentences, possibly with a "Read more" with a link to the Wikipedia-page.

The initial idea for the plug-in was that it should recognize entities in general, but during development I narrowed the scope to only include Person-entities. It should not be difficult, though, to extend it to support other entities as well. During the first round of usability testing my sketches included Place-entities, and thus there's already data on what information the journalists need on countries and cities.

I would also recommend downloading DBpedia Spotlight and the knowledge bases locally to avoid any downtime issues. It might also be worth looking into alternatives to DBpedia Spotlight, as it's not suited for Norwegian input, or look out for updates that support this.

Finally, I hope that this thesis can be of use to VG. Like most employees in VG I'm keenly aware that tagging and information architecture is of huge importance the company, both VG and Schibsted. Linked Open

Data is still a small field, but I've attempted to give an overview of the possibilities that lie there, and hopefully VG will now be able to weigh the advantages against the challenges. I look forward to seeing the future developments in the Linked Open Data field, and the evolving of VG's information architecture, whether it's with Linked Open Data or not.

Appendices

Appendix A

Interview guide for exploratory research

- Informert samtykke
- Forklare hvem jeg er, hva jeg skal spørre om, hva jeg skal bruke informasjonen til, hvor lang tid intervjuet og observasjonen vil ta osv..
- Forklare formålet mitt: Vil prøve å få innblikk i prosessen med research rundt en sak, særlig når man har begynt med selve skrivingen av artikkelen. Jeg har ingen forkunnskaper om dette, så spørsmålene er ganske grunnleggende. Skal intervjue først, deretter observere, så intervjuet er kun for å få litt klarhet så jeg forstår hva jeg ser under observasjonen

1. Hvilken avdeling jobber du for?
2. Hvordan er prosessen for deg fra du får saken til den er ferdig?
3. Pleier du å gjøre noe research utenfor primærkilden til saken?
4. Skriver du saker direkte inn i DrPublish, eller bruker du et annet skriveprogram først?
5. Hvilke online kilder bruker du? Stoler du på f.eks. Wikipedia som pålitelig kilde?
6. Hvis du skal sjekke noe utenfor primærkilden, f.eks. NTB, hva pleier du å sjekke?

HUSKELISTE:

- Penn
- Notatblokk
- Informert samtykke-skjema

Appendix B

Guide for formative usability testing

- Informert samtykkeskjema
- Tester prototypen, ikke deg. Ikke vær redd for å gi kritikk.
- Forklare hva plug-inen skal gjøre
- **Startskjerm:** Du har skrevet denne teksten om Obama, men ønsker å finne ut hvor gammel han er. Du vil bruke denne modulen, du finner den i listen, og trykker på den, og deretter kommer dette skjermbildet opp. Hva vil du gjort så?

  Hva tenker du at skjer når du trykker på den knappen?
- **Søkeresultater:** Da kommer dette opp. Hva ville du gjort så?

  Hva slags informasjon tror du kommer opp?
- **Faktaboks Obama:** Forklare. Er dette nyttig informasjon, gitt at den er korrekt?

  Er det noe av informasjonen som vises som du ikke kan tenke deg at vil bli brukt?
- **Faktaboks Tyskland:** Forklare. Er dette nyttig informasjon, gitt at den er korrekt?

  Er det noe av informasjonen som vises som du ikke kan tenke deg at vil bli brukt?
- Er denne modulen noe du kunne brukt selv?
- Navnforslag?

Appendix C

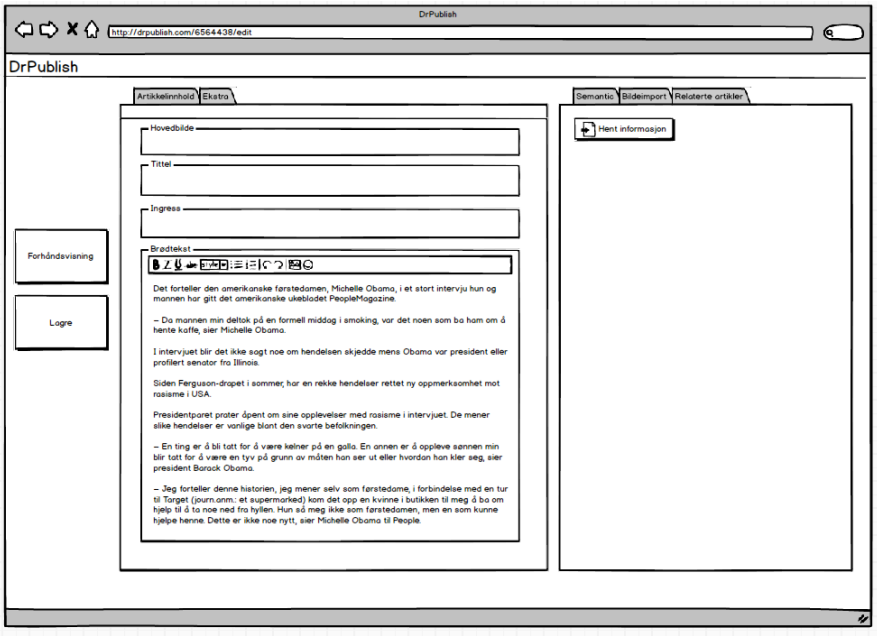
Guide to final usability test – Thinking Aloud

- Informert samtykke
- Spesifisere at jeg tester prototypen, ikke deltakeren
- Be journalisten "tenke høyt", ikke nødvendigvis forklare hvorfor hun/han gjør hva de gjør
- Scenarioer: Barack Obama, Brad Pitt, Vladimir Putin
- Jeg noterer hva de sier og hva de gjør
- Hjelper hvis de møter på bugs, men prøver å ikke blande meg inn
- Spørsmål:
  - Hvordan opplever du å bruke denne sammenlignet med måten du vanligvis finner den type informasjon?
  - Er dette noe du kunne tenkt deg å bruke til vanlig, gitt at den var bug-fri?
  - Hvis jeg sier at informasjonen er hentet fra Wikipedia, stoler du fortsatt på den?
  - Hva liker du?
  - Hva liker du ikke?

Appendix D

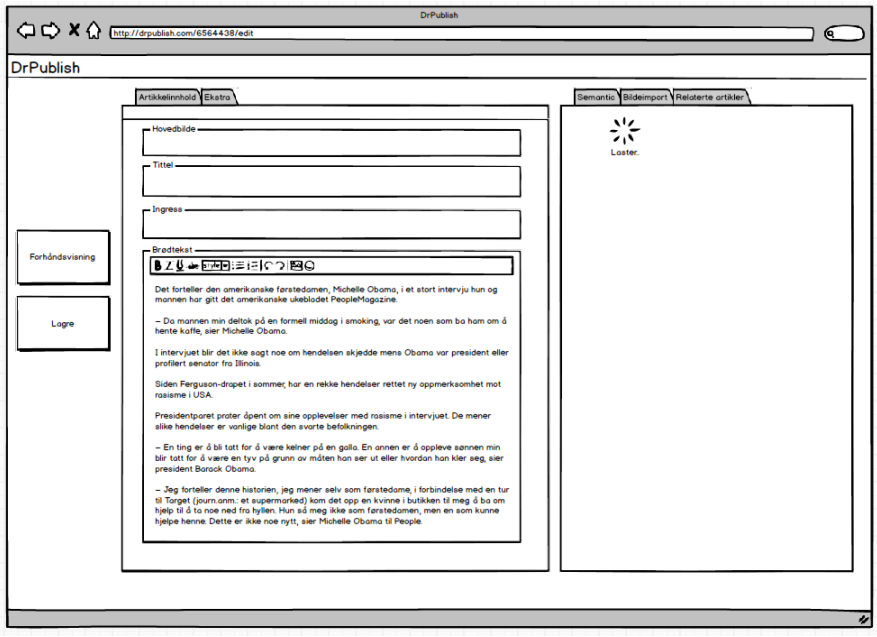
Low-fidelity prototype

Figure D.1: Low-fidelity prototype: The start screen



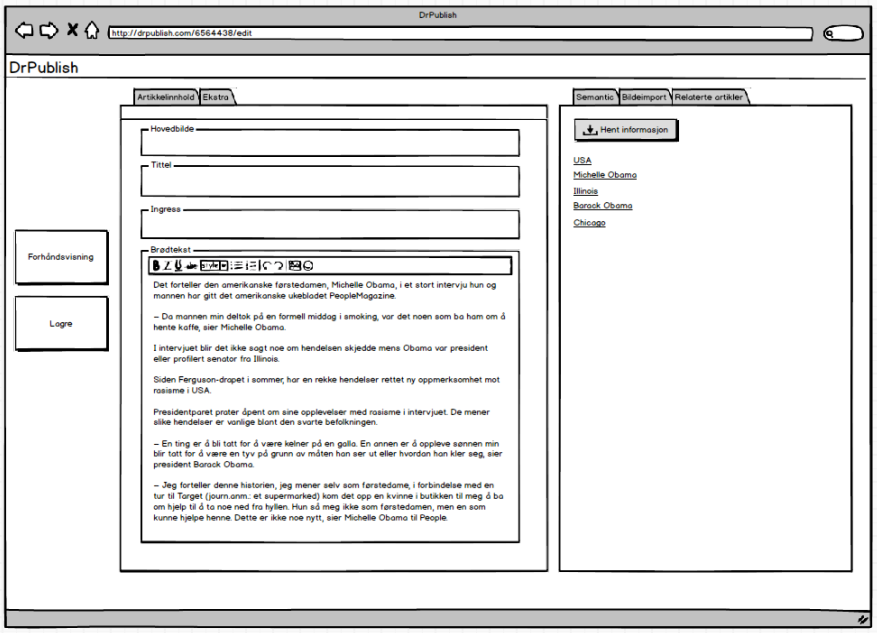
Once the plug-in has been loaded into the user interface it displays a button called "Get information".

Figure D.2: Low-fidelity prototype: The loading screen



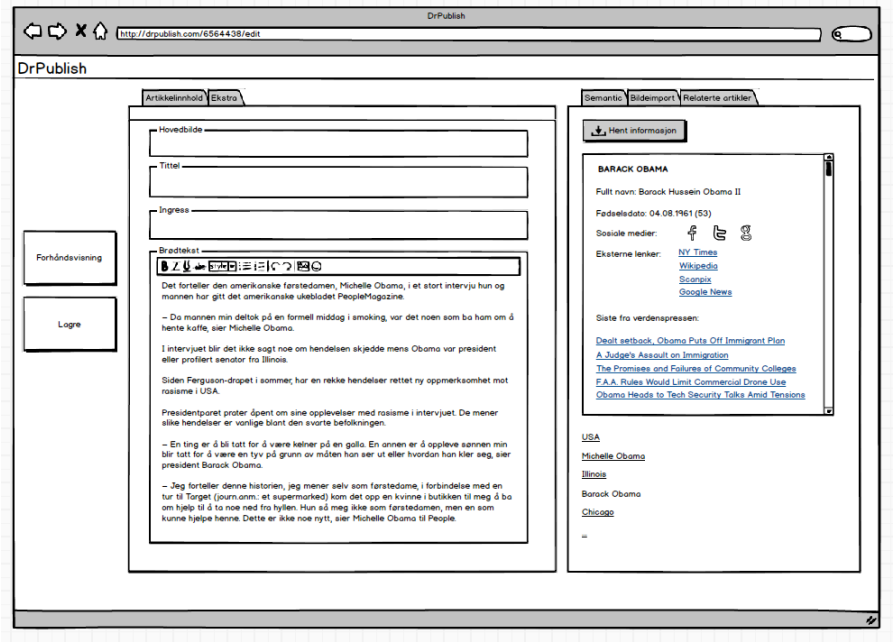
Once the "Get information"-button has been clicked, it displays a loading icon while it retrieves entities in the text.

Figure D.3: Low-fidelity prototype: The search results



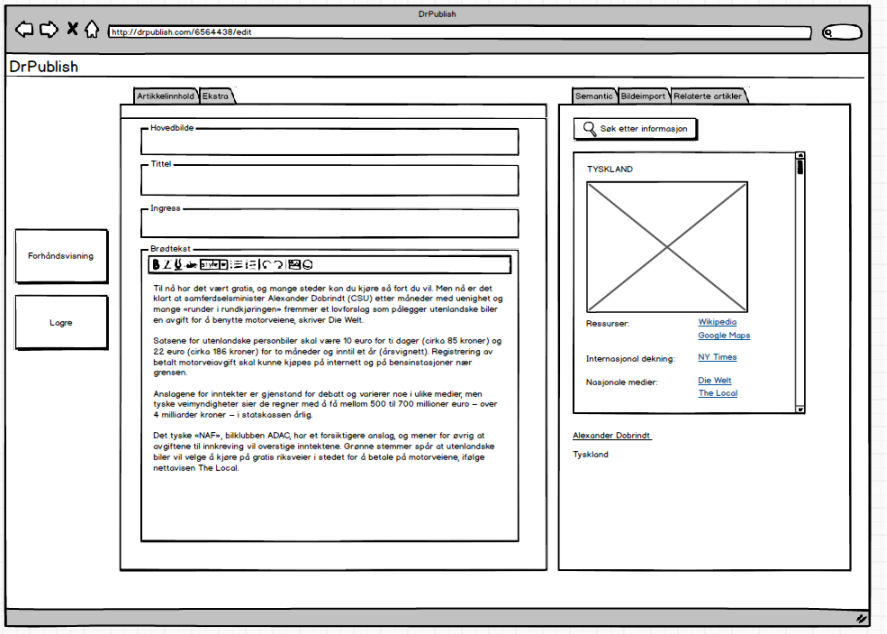
The plug-in has found multiple search results, and displays them as a list. Each list item is clickable to load information on each entity.

Figure D.4: Low-fidelity prototype: Displaying information on a person



By clicking on an entity, in this case "Barack Obama", the plug-in will display the full name, date of birth, age, social media accounts and various links, including ones to the most recent articles by Ny Times on the subject.

Figure D.5: Low-fidelity prototype: Displaying information on a country



The infobox on countries contains a map, links to local newspapers, and to Wikipedia, NY Times and Google Maps. Unfortunately I was not able to implement this functionality into the final result.

Bibliography

Aprosio, Alessio Palmero, Claudio Giuliano and Alberto Lavelli (2013). 'Automatic Mapping of Wikipedia Templates for Fast Deployment of Localised DBpedia Datasets'. In: *Proceedings of the 13th International Conference on Knowledge Management and Knowledge Technologies*. i-Know '13. Graz, Austria: ACM, 1:1–1:8.

Berners-Lee, Tim, James Hendler and Ora Lassila (2001). 'The Semantic Web'. In: *The Scientific American* 284.5, pp. 34–43.

Bocconi, Stefano and Angela Fogarolli (2010). 'Enriching a News Portal with Semantic Information: An Entity-Based Approach'. In: *AAAI Spring Symposium: Linked Data Meets Artificial Intelligence'10*.

Boren, T. and J. Ramey (2000). 'Thinking aloud: reconciling theory and practice'. In: *Professional Communication, IEEE Transactions on* 43.3, pp. 261–278.

Charmaz, Kathy (2005). 'Grounded Theory in The 21st Century'. English. In: *The Sage Handbook of Qualitative Research*. Sage Publications, pp. 507–535.

Cozby, Paul C. (2008). *Methods in Behavioral Research*. 10th ed. McGraw-Hill, pp. 100–115.

Dumas, J. and J. Fox (2007). 'Usability testing: Current practice and future directions'. In: *The Human Computer Interaction Handbook 2nd ed.* Human factors and ergonomics. Boca Ratin, Fla: CRC Press, pp. 1129–1143.

Gangemi, Aldo (2013). 'A Comparison of Knowledge Extraction Tools for the Semantic Web'. English. In: *The Semantic Web: Semantics and Big Data*. Ed. by Philipp Cimiano et al. Vol. 7882. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 351–366.

Gerber, Elizabeth M. and Julie Hui (2013). 'Crowdfunding: Motivations and Deterrents for Participation'. In: *ACM Trans. Comput.-Hum. Interact.* 20.6, 34:1–34:32.

Hitzler, Pascal, Markus Krötzsch and Sebastian Rudolph (2009). *Foundations of Semantic Web Technologies*. Chapman & Hall/CRC.

Howe, Jeff (2008). *Crowdsourcing: Why the Power of the Crowd Is Driving the Future of Business*. 1st ed. New York, NY, USA: Crown Publishing Group.

Jørgensen, Anker H. (1990). 'Thinking-aloud in user interface design: a method promoting cognitive ergonomics'. In: *Ergonomics* 33.4, pp. 501–507.

Kittur, Aniket and Robert E. Kraut (2008). 'Harnessing the Wisdom of Crowds in Wikipedia: Quality Through Coordination'. In: *Proceedings of*

the 2008 ACM Conference on Computer Supported Cooperative Work. CSCW '08. San Diego, CA, USA: ACM, pp. 37–46.

Kobilarov, Georgi et al. (2009). 'Media Meets Semantic Web – How the BBC Uses DBpedia and Linked Data to Make Connections'. English. In: *The Semantic Web: Research and Applications*. Vol. 5554. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 723–737.

Lazar, Jonathan, Junjuan Heidi Feng and Harry Hochheiser (2010). *Research Methods in Human-Computer Interaction*. 1st ed. Wiley.

Mahdisoltani, Farzaneh, Joanna Biega and Fabian M Suchanek (2015). 'YAGO3: A Knowledge Base from Multilingual Wikipedias'. In: *Conference on Innovative Data Systems Research 2015*.

Maxwell, Joseph A. (2005). *Qualitative Research Design: an Interactive Approach*. 2nd ed. Sage Publications, pp. 95–104.

Morville, Peter and Louis Rosenfeld (2006). *Information Architecture on the World Wide Web*. 3rd ed. O'Reilly Media, pp. 77–81.

Raimond, Yves et al. (2010). 'Use of Semantic Web technologies on the BBC Web Sites'. English. In: *Linking Enterprise Data*. Ed. by David Wood. Springer US, pp. 263–283.

Strauss, Anselm L. (1987). *Qualitative Analysis for Social Scientists*. Cambridge University Press.

Voigt, Martin, Michael Aleythe and Peter Wehner (2013). 'Towards Topics-based, Semantics-assisted News Search'. In: *Proceedings of the 3rd International Conference on Web Intelligence, Mining and Semantics*. WIMS '13. Madrid, Spain: ACM, 6:1–6:7.

Zaveri, Amrapali et al. (2013). 'User-driven Quality Evaluation of DBpedia'. In: *Proceedings of the 9th International Conference on Semantic Systems*. I-SEMANTICS '13. Graz, Austria: ACM, pp. 97–104.

Links to bibliography

Aprosio, Giuliano and Lavelli 2013:
http://doi.acm.org/10.1145/2494188.2494196

Berners-Lee, Hendler and Lassila 2001:
http://www.cs.umd.edu/ golbeck/LBSC690/SemanticWeb.html

Bocconi and Fogarolli 2010:
https://www.aaai.org/ocs/index.php/SSS/SSS10/paper/view/1123

Gangemi 2013:
http://dx.doi.org/10.1007/978-3-642-38288-8_24

Gerber and Hui 2013:
http://doi.acm.org/10.1145/2530540

Jørgensen 1990:
http://dx.doi.org/10.1080/00140139008927157

Kittur and Kraut 2008:
http://doi.acm.org/10.1145/1460563.1460572

Kobilarov et al. 2009:
http://dx.doi.org/10.1007/978-3-642-02121-3_53

Mahdisoltani, Biega and Suchanek 2015:
http://suchanek.name/work/publications/cidr2015.pdf

Raimond et al. 2010:
http://dx.doi.org/10.1007/978-1-4419-7665-9_13

Voigt, Aleythe and Wehner 2013:
http://doi.acm.org/10.1145/2479787.2479822

Zaveri et al. 2013:
http://doi.acm.org/10.1145/2506182.2506195